A knowledge-based approach to ontology learning and semantic annotation

Roberto Navigli, Paola Velardi

[1] Dipartimento di Informatica, Università di Roma La sapienza, Italy

Abstract. The so-called Semantic Web vision will certainly benefit from automatic semantic annotation of words in documents. We present a method, called *structural semantic interconnections (SSI)*, that creates structural specifications of the possible senses for each word in a context, and selects the best hypothesis according to a grammar *G*, describing relations between sense specifications. The method has been applied to different semantic disambiguation problems, like automatic ontology construction, sense-based query expansion, disambiguation of words in glossary definitions. Evaluation experiments have been performed on each disambiguation task, as well

1 Introduction

Word sense disambiguation (WSD) is perhaps the most critical task in the area of computational linguistics. We do not attempt here a survey of the field, but we refer the interested reader to the Senseval[1] home page for a collection of state of art sense disambiguation methods, and the results of public challenges in this area. During the most recent Senseval evaluation, the best system in the English all-words task [1] reached a 69% precision and recall, a performance that in [2] is claimed to be well below the threshold that produces improvements in a text retrieval task.

The lack of high-performing methods for sense disambiguation may be considered the major obstacle that prevented an extensive use of natural language processing techniques in many areas of information technology, such as information classification and retrieval, query processing, advanced web search, document warehousing, etc. In these application fields, the use of statistical and algebraic methods largely prevails on knowledge-based methods, a tendency that clearly emerges in main information retrieval conferences and challenges, such as SIGIR[2] and TREC[3].

However, emerging applications like the so called Semantic Web [3] foster "*an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation*", an objective that hardly will be met through mere manual semantic annotations using languages such as XML, RDF, OWL etc [4]. Large-scale semantic annotation projects would greatly benefit from reliable methods for automatic sense selection. In recent years, the results

[1] http://www.senseval.org/

[2] http://www.acm.org/sigir/

[3] http://trec.nist.gov/

of many research efforts for the construction of on-line knowledge repositories, ontologies and glossaries are now available (e.g. [5] [6] [7]), creating new opportunities for knowledge-based sense disambiguation methods.
In this paper we present a WSD algorithm, called *structural semantic interconnections (SSI)*, that uses graphs to describe the objects to analyze (word senses) and a context-free grammar to detect relevant semantic patterns between graphs. Sense classification is based on the number and type of detected interconnections. The graph representation of word senses is automatically built from several available resources, such as lexicalized ontologies, annotated corpora, and glossaries. The SSI algorithm is applied to different problems, namely:

§ extending and trimming a general-purpose ontology with complex domain terms (e.g. respectively for computer networks, tourism, and finance: *local area networks, hotel facility, leveraged buy-out*).
§ disambiguating the words in a natural language concept definition: this is a prerequisite for creating formal axiomatic sense descriptions from informal ones;
§ disambiguating the words in a query for sense-based web query expansion.

The paper is organized as follows: in Section 2 we the structural semantic interconnection algorithm and describes the context-free grammar for detecting semantic interconnections. Section 3 provides implementation details for the three word sense disambiguation problems listed above. Finally, Section 4 is dedicated to the description of several experiments that we made on standard and ad-hoc testing environments.

2 The SSI algorithm for sense tagging

Our approach to WSD lies in the *structural pattern recognition* framework. Structural or syntactic pattern recognition [8] [9] has proven to be effective when the objects to be classified contain an inherent, identifiable organization, such as image data and time-series data. For these objects, a representation based on a "flat" vector of features causes a loss of information that negatively impacts on classification performances. Word senses clearly fall under the category of objects that are better described through a set of structured features.
The classification task in a structural pattern recognition system is implemented through the use of grammars that embody precise criteria to discriminate among different classes. Learning a structure for the objects to be classified is often a major problem in many application areas of structural pattern recognition. In the field of computational linguistics, however, several efforts have been made in the past years to produce large lexical knowledge bases and annotated resources, offering an ideal starting point for constructing structured representations of word senses.

2.1 Building structural representations of word senses

A structural representation of word senses is automatically built using a variety of knowledge sources, i.e. WordNet, Domain Labels [12] annotated corpora like SemCor and *LDC-DSO*[4]. We use this information to automatically generate labeled directed

[4] LDC http://www.ldc.upenn.edu/

graph (*digraph*) representations of word senses. We call these *semantic graphs*, since they represent alternative conceptualizations for a lexical item.
Figure 1 shows an example of the semantic graph generated for senses #1 of *market*, where nodes represent concepts (WordNet synsets), and edges are semantic relations. In each graph, we include only nodes with a maximum distance of 3 from the central node, as suggested by the dashed ovals in Figure 1. This distance has been experimentally established.

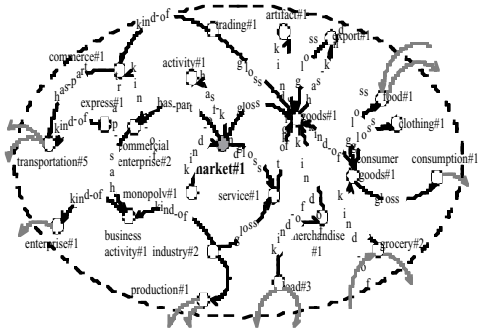

Figure 1. Graph representations for sense #1 of *market*.

All the used semantic relations are explicitly encoded in WordNet, except for the latter three. *Topic*, *gloss* and *domain* are extracted respectively from annotated corpora, sense definitions and domain labels.

2.2 Summary description of the SSI algorithm

The SSI algorithm consists of an initialization step and an iterative step.
In a generic iteration of the algorithm the input is a list of co-occurring terms $T = [\ t_1, \ldots, t_n\ ]$ and a list of associated senses $I = [S^{t1},\ldots,S^{tn}]$, i.e. the semantic interpretation of T, where S^{ti} [5] is either the chosen sense for t_i (i.e., the result of a previous disambiguation step) or the empty set (i.e., the term is not yet disambiguated).
A set of *pending* terms is also maintained, $P = \{\ t_i \mid S^{ti} = \varnothing\ \}$. I is named the *semantic context* of T and is used, at each step, to disambiguate new terms in P.
The algorithm works in an iterative way, so that at each stage either at least one term is removed from P (i.e., at least a pending term is disambiguated) or the procedure stops because no more terms can be disambiguated. The output is the updated list I of senses associated with the input terms T.
Initially, the list I includes the senses of monosemous terms in T. If no monosemous terms are found, the algorithm makes an initial guess based on the most probable sense of the less ambiguous term. The initialisation policy is adjusted depending upon

[5] Note that with we refer interchangeably to the semantic graph associated with a sense or to the sense *name*.

the specific WSD task considered. Section 3 describes the policy adopted for the task of gloss disambiguation in WordNet.

During a generic iteration, the algorithm selects those terms t in P showing an interconnection between at least one sense S of t and one or more senses in I. The likelihood for a sense S of being the correct interpretation of t, given the semantic context I, is estimated by the function $f_I : C \to \Re$, where C is the set of all the concepts in the ontology O, defined as follows:

$$f_I(S,t) = \begin{cases} \rho(\{\phi(S,S') \mid S' \in I\}) & \text{if } S \in Senses(t) \subset Synsets \\ 0 & \text{otherwise} \end{cases}$$

where *Senses(t)* is the subset of concepts C in O associated with the term t, and $\varphi(S,S') = \rho'(\{w(e_1 \cdot e_2 \cdot \ldots \cdot e_n) \mid S \xrightarrow{e_1} S_1 \xrightarrow{e_2} \ldots \xrightarrow{e_{n-1}} S_{n-1} \xrightarrow{e_n} S'\})$, i.e. a function (ρ') of the weights (w) of each path connecting S with S', where S and S' are represented by semantic graphs. A semantic path between two senses S and S', $S \xrightarrow{e_1} S_1 \xrightarrow{e_2} \ldots \xrightarrow{e_{n-1}} S_{n-1} \xrightarrow{e_n} S'$, is represented by a sequence of edge labels $e_1 \cdot e_2 \cdot \ldots \cdot e_n$. A proper choice for both ρ and ρ' may be the *sum* function (or the *average sum* function).

A context-free grammar $G = (E, N, S_G, P_G)$ encodes all the meaningful semantic patterns. The terminal symbols (E) are edge labels, while the non-terminal symbols (N) encode (sub)paths between concepts; S_G is the start symbol of G and P_G the set of its productions.

We associate a weight with each production $A \to \alpha$ in P_G, where $A \in N$ and $\alpha \in (N \cup E)^*$, i.e. α is a sequence of terminal and non-terminal symbols. If the sequence of edge labels $e_1 \cdot e_2 \cdot \ldots \cdot e_n$ belongs to $L(G)$, the language generated by the grammar, and provided that G is not ambiguous, then $w(e_1 \cdot e_2 \cdot \ldots \cdot e_n)$ is given by the sum of the weights of the productions applied in the derivation $S_G \Rightarrow^+ e_1 \cdot e_2 \cdot \ldots \cdot e_n$. The grammar G is described in the next section.

Finally, the algorithm selects $\arg\max_{S \in C} f_I(S)$ as the most likely interpretation of t and updates the list I with the chosen concept. A threshold can be applied to $f_I(S)$ to improve the robustness of system's choices.

At the end of a generic iteration, a number of terms is disambiguated and each of them is removed from the set of pending terms P. The algorithm stops with output I when no sense S can be found for the remaining terms in P such that $f_I(S) > 0$, that is, P cannot be further reduced.

In each iteration, interconnections can only be found between the sense of a pending term t and the senses disambiguated during the previous iteration.

A special case of input for the SSI algorithm is given by $I = [\varnothing, \varnothing, \ldots, \varnothing]$, that is when no initial semantic context is available (there are no monosemous words in T). In this case, an initialisation policy selects a term $t \in T$ and the execution is forked into as many processes as the number of senses of t.

2.3 The grammar

The grammar *G* has the purpose of describing meaningful interconnecting patterns among semantic graphs representing conceptualisations in O. We define a *pattern* as a sequence of *consecutive* semantic relations $e_1 \cdot e_2 \cdot \ldots \cdot e_n$ where $e_i \in E$, the set of terminal symbols, i.e. the vocabulary of conceptual relations in O. Two relations e_i e_{i+1} are consecutive if the edges labelled with and e_{i+1} are incoming and/or outgoing from the same concept node, that is $\overset{e_i}{\rightarrow}_{(S)} \overset{e_{i+1}}{\rightarrow}$, $\overset{e_i}{\leftarrow}_{(S)} \overset{e_{i+1}}{\rightarrow}$, $\overset{e_i}{\rightarrow}_{(S)} \overset{e_{i+1}}{\leftarrow}$, $\overset{e_i}{\leftarrow}_{(S)} \overset{e_{i+1}}{\leftarrow}$. A meaningful pattern between two senses *S* and *S'* is a sequence $e_1 \cdot e_2 \cdot \ldots \cdot e_n$ that belongs to $L(G)$.

In its current version, the grammar *G* has been defined manually, inspecting the intersecting patterns automatically extracted from pairs of manually disambiguated word senses co-occurring in different domains. Some of the rules in *G* are inspired by previous work on the eXtended WordNet project described in (Milhalcea and Moldovan, 2001). The terminal symbols e_i are the conceptual relations extracted from WordNet and other on-line lexical-semantic resources, as described in Section 2.1.

G is defined as a quadruple (E, N, S_G, P_G), where E = { $e_{\text{kind-of}}$, $e_{\text{has-kind}}$, $e_{\text{part-of}}$, $e_{\text{has-part}}$, e_{gloss}, $e_{\text{is-in-gloss}}$, e_{topic}, … }, N = { S_G, S_s, S_g, S_1, S_2, S_3, S_4, S_5, S_6, E_1, E_2, … }, and P_G includes about 50 productions.

As stated in previous section, the weight $w(e_1 \cdot e_2 \cdot \ldots \cdot e_n)$ of a semantic path $e_1 \cdot e_2 \cdot \ldots \cdot e_n$ is given by the sum of the weights of the productions applied in the derivation $S_G \Rightarrow^+ e_1 \cdot e_2 \cdot \ldots \cdot e_n$. These weights have been learned using a *perceptron* model, using standard word sense disambiguation data, such as the SemCor corpus.

Examples of the rules in G are provided in the subsequent Section 3.

3 Three applications of the SSI algorithm

The SSI algorithm has been applied in three different WSD tasks. Each of these tasks required some adaptation of the algorithm, as detailed hereafter.

3.1 Disambiguation of textual definitions in an ontology or glossary.

Glossaries and ontologies often provide a textual definition of concepts in which words are left ambiguous. For example, the WordNet definition of *transport#3* is "*the commercial enterprise of transporting goods and materials*". In this gloss, the word *enterprise* has 3 senses, and *material* has 6. Associating the correct sense with each word in a gloss is a sort of preliminary step to construct formal concept definitions from informal ones [13].

For the gloss disambiguation task, the SSI algorithm is initialized as follows: In step 1, the list *I* includes the synset *S* whose gloss we wish to disambiguate, and the list *P* includes all the terms in the gloss and in the gloss of the hyperonym of *S*. Words in the hyperonym's gloss are useful to augment the context available for disambiguation.

In the following, we present a sample execution of the SSI algorithm for the gloss disambiguation task applied to sense #1 of *retrospective*: "*an exhibition of a representative selection of an artist's life work*". For this task the algorithm uses a

context enriched with the definition of the synset hyperonym, i.e. *art exhibition#1*: "*an exhibition of art objects (paintings or statues)*".
Initially we have:

I = { *retrospective#1* }[6]
P = { *work, object, exhibition, life, statue, artist, selection, representative, painting, art* }

At first, *I* is enriched with the senses of monosemous words in the definition of *retrospective*#1 and its hyperonym:

I = { *retrospective#1, statue#1, artist#1* }
P = { *work, object, exhibition, life, selection, representative, painting, art* }

since *statue* and *artist* are monosemous terms in WordNet. During the first iteration, the algorithm finds three matching paths[7]:
retrospective#1 $\xrightarrow{kind-of}{}^{2}$ *exhibition#2*, *statue#1* $\xrightarrow{kind-of}{}^{3}$ *art#1* and *statue#1* $\xrightarrow{kind-of}{}^{6}$ *object#1* (all instances of rule S_1 in Figure 4).

This leads to:

I = { *retrospective#1, statue#1, artist#1, exhibition#2, object#1, art#1* }
P = { *work, life, selection, representative, painting* }

During the second iteration, a hyponymy/holonymy path (rule S_2) is found:
art#1 $\xrightarrow{has-kind}{}^{2}$ *painting#1* (painting is a kind of art)
which leads to:
I = { *retrospective#1, statue#1, artist#1, exhibition#2, object#1, art#1, painting#1* }
P = { *work, life, selection, representative* }

The third iteration finds a co-occurrence (topic rule S_5 in Figure 4) path between *artist#1* and sense 12 of *life* (*biography, life history*):
artist#1 $\xrightarrow{topic}$ life#12
then, we get:
I = { *retrospective#1, statue#1, artist#1, exhibition#2, object#1, art#1, painting#1, life#12* }
P = { *work, selection, representative* }

The algorithm stops because no additional matches are found. The chosen senses concerning terms contained in the hyperonym's gloss were of help during disambiguation, but are now discarded. Thus we have:

GlossSynsets(retrospective#1) = { *artist#1, exhibition#2, life#12, work#2* }

---

[6] For convenience here we denote *I* as a set rather than a list.
[7] With *S* $\xrightarrow{R}{}^{i}$ *S'* we denote a path of *i* consecutive edges labeled with the relation *R* interconnecting *S* with *S'*.

In Section 5 an evaluation of this task is presented.

3.2 Association of complex concepts to multi-word terms.

Complex terms can often be defined *compositionally* from its constituents. For example, we can define *revenue management* as "*the act of managing government income due to taxation*", a definition obtained through the concatenation of the appropriate definitions of *management* and *revenue*. Complex concepts are included in general purpose ontologies to a limited extent, and domain ontologies are available only in few technical fields (e.g. medicine, computer science).
Our aim was to extend an existing general purpose ontology through the semantic interpretation of multi-word terms. For example, *revenue* and *management* are in WordNet, but not *revenue management* as a complex concept. Both *revenue* and *management* have two senses in WordNet, which leads to 4 possible compositional semantic interpretations[8] for the complex term. In order to extend with complex domain concepts a general-purpose ontology we devised an ontology learning system, called OntoLearn [15][16], which is again based on the SSI algorithm. With reference to the above example, the objective of Ontolearn is to select the appropriate senses (*management*#1 and *revenue*#2) and attach the new concept under its direct hyperonym in WordNet (i.e. *management*#1). The first step of the Ontolearn procedure is to automatically extract from dedicated corpora a list of multi-word terms in a specific domain (we experimented on finance, tourism and computer networks). Terms are then arranged in tree structures, based on simple string inclusion. An example is in Figure 2, obtained from a tourism domain.

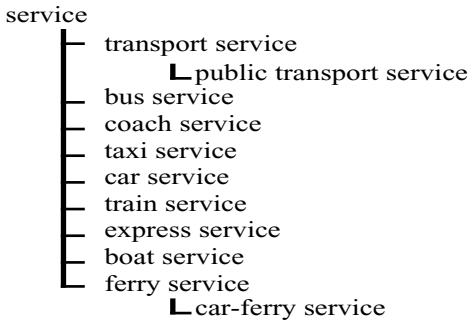

Figure 2. Complex terms arranged in a tree-structure rooted in *service*.

The tree is used as the initial context to activate semantic disambiguation. With reference to Figure 5, the list *T* in input to the SSI algorithm would be: [*service, train, ferry, car, car-ferry, boat, bus, coach, transport, public, taxi, express*]. The initialization of *I* in this case is more complex. If the context is rich enough, as in the *service* example, one or more monosemous terms are usually found to start the

[8] Compositional interpretation of complex nominals applyes in many cases, but clearly not in all. A backoff strategy is to parse glossary definitions, a strategy whose description is outside the scope of this paper.

disambiguation process. For example, *taxi* and *car-ferry* have only one sense (respectively, "*a car driven by a person whose job is to take passengers where they want to go in exchange for money*" and "*a ferry that transports motor vehicles*"), and several interconnections are found during the first execution of the iterative step between the semantic graphs of the monosemous words and the semantic graphs corresponding to the correct sense of the other words in *T*.
For example, the following rules allow it to disambiguate *car, ferry, boat* and *coach* during the first iteration of the SSI algorithm:

$taxi\#1 \xrightarrow{kind-of} car\#1$ *(hyper rule S_1)*

$taxi\#1 \xrightarrow{kind-of} car\#1 \xleftarrow{kind-of} coach\#3$ *(hyper/hypo rule S_3)*

$taxi\#1 \xrightarrow{gloss} passenger\#1 \xleftarrow{gloss} coach\#3$ *(gloss rule S_6)*

$car-ferry\# \xrightarrow{kind-of} ferry\#1$ *(hyper rule S_1)*

$car-ferry\# \xrightarrow{kind-of} ferry\#1 \xleftarrow{has-kind} boat\#1$ *(hyper/hypo rule S_3)*

The subsequent iterations produce a complete disambiguation of the list *T*.
If no monosemous words are found, we explore two alternatives: either we provide manually the synset of the root term (*service*#2: "*a company or agency that performs a public service*") or we fork the execution of the algorithm in as many processes as the number of senses of the root term *h*. Let *n* be such a number. For each process *i* (*i* = 1, …, *n*), the input is given by $I_i = \{\ldots, S_h^i, \ldots\}$ where S_h^i is the *i*-th sense of *h* in *Senses(h)*. Each execution outputs a (partial or complete) semantic context I_i. Finally, the most likely context I_m is obtained by choosing:

$$m = \arg\max_{1 \le i \le n} \sum_{S \in I_i} f_{I_i}(S).$$

The intuition is that, since the root term is included in all the nodes of a term-tree, its "correct" sense should exhibit a high degree of semantic interconnectivity with all the other components. This means that, eventually, one of the processes should cumulate a higher weight $\sum_{S \in I_i} f_{I_i}(S)$ with respect to the others.
When the correct senses have been eventually determined for all the term components of a term-tree, the tree is re-arranged using taxonomic relations, as shown in Figure 3. More details and evaluations on the OntoLearn system are provided in the aforementioned papers. A summary evaluation is in Section 4.

3.3 Sense-based query expansion

The third application that we explored is the use of sense information for query expansion. Sense-based query expansion has not really proven its effectiveness in information retrieval applications: published results are not very recent [17] [18], and the use of statistical methods is currently prevailing. A more recent work [2] analyzes the effect of expanding a query with WordNet synsets, in a "canned" experiment where all words are manually disambiguated. In this paper it is shown that a substantial increase in performance is obtained only with less than 10% errors in the word sense

disambiguation (WSD) task. Unfortunately, the state of affairs for WSD [1] is far below this threshold, as remarked in the introduction.

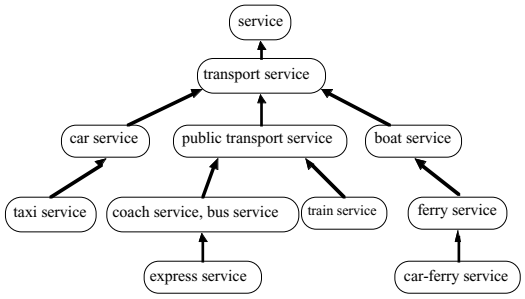

Figure 3. Complex concepts arranged in a taxonomic structure rooted in *service#2*.

However, the 10% error-rate threshold established in [2] is based on a query expansion technique that associates with each disambiguated term the list of its synonyms and/or hyponyms, in an and-or fashion. To our knowledge, all sense-based expansion algorithms in literature use basically "kind-of" and synonymy information to generate the expanded query.

Our claim is that the SSI algorithm offers the additional opportunity of expanding a query with the concepts in an interconnection pattern; for example, the interconnection pattern between *smoke* and *disease* in "diseases caused by smoke?"[9]:

$$smoke\#7 \xrightarrow{gloss} tobacco\#1 \xrightarrow{kind-of} drug\#1 \xrightarrow{domain} disease\#1$$

suggests an expansion with the words *tobacco* and *drug*.

In a query expansion task, the SSI algorithm receives a list T including the set of keywords in a query. For example, the (unexpanded) query above is rewritten as T = [*disease*, *cause*, *smoke*].

The problem here is that real-world queries are usually very short, reducing the probability of finding monosemous terms[10] for initializing the semantic context I. The SSI process in this case is forked in as many executions as the number of senses of the less ambiguous term. Given the limited and less focused context provided by short queries, the precision in disambiguation is expected to be lower than for the applications described in previous sub-sections. However, the evaluation section shows that, even in presence of a WSD error rate higher than the 10% threshold estimated in [2], a sense-based expansion strategy based on non-taxonomic relations seems very promising.

[9] This query is taken from those used in the public challenge TREC 2002 (web query track: http://www.ted.cmis.csiro.au/TRECWeb/guidelines_2002.html)

[10] The example above is indeed fortunate since *disease* has only one sense.

4 Experimental results

This section is organized as follows: first, we provide evaluations for each of the three applications described in previous Section. Then, we describe an evaluation experiment over a standard WSD dataset, the so-called "all-words" dataset of Senseval-2.
To evaluate the performance of the various disambiguation tasks we used two common measures: *recall* and *precision*. *Recall* provides the percentage of right senses with respect to the overall number of terms to be disambiguated. In fact, when the disambiguation algorithm terminates, the list *P* may still include terms for which no relation with the synsets in *I* could be found. Furthermore, the number of disambiguated senses depends on the weights associated with the rules in *G*. *Precision* measures the percentage of right senses with respect to the retrieved senses. Notice that thresholding the measure of certainty $\zeta_{I_i}(S)$ in SSI algorithms (Section 3.1) can optimally tune precision and recall: in our view, a higher precision is often more important than a high recall, especially in the context of specific sense-based tasks, like query expansion, where even few errors may produce very negative results on retrieval. Therefore, we pursued high precision and "reasonable" recall. The balance between these two measures has been tuned for each specific disambiguation task.
Furthermore, in our experiments, we also computed a baseline precision, using the "first sense choice" heuristic. In WordNet, synsets are ordered by probability of use, i.e. the first synset is the most likely sense. For a fair comparison, the baseline is computed only on the words for which the algorithm could retrieve a synset.

4.1 Evaluation of gloss disambiguation task

The gloss disambiguation algorithm has been evaluated on two sets of glosses: a first set of 100 general-purpose glosses[11] and a second set of 305 glosses from a tourism domain. This allows us to evaluate the method both on a restricted domain and a non-specialized task.
For each word in a gloss, the appropriate WordNet sense has been manually assigned, for over 1,000 words. The annotations are in part those made available by the authors of the already mentioned eXtended WordNet project [14], in part (e.g. those in the tourism domain) have been assigned by the three authors of the research described in [13].
Table 1 gives an overview of the results. Table 1a provides an overall evaluation of the algorithm, while table 1b computes precision and recall grouped by morphological category. The precision is quite high (well over 90% for both general and domain glosses) but the recall is around 40%. Remarkably, the achieved improvement in precision with respect to the baseline is much higher for general glosses than for domain glosses. This is motivated by the fact that general glosses include words that are more ambiguous than those in domain glosses. Therefore, the general gloss baseline is quite low. This means also that the disambiguation task is far more complex in the case of general glosses, where our algorithm shows particularly good performance.
An analysis of performance by morphological category (Table 1b) shows that noun disambiguation has much higher recall and precision. This is motivated by the fact

[11] The 100 generic glosses have been randomly selected among the 809 glosses used in the OntoClean project (see [13] for details)

that, in WordNet, noun definitions are richer than those of verbs and adjectives. The WordNet hierarchy for verbs is known as being more problematic with respect to nouns. As already mentioned, we plan to integrate in our algorithm verb information from FrameNet and VerbNet, lexico-semantic knowledge bases, providing rich information especially for verbs.

Domains		# glosses	# words	# disamb. words	# of which ok	Recall	Precision	Baseline Precision
Tourism		305	1345	636	591	47,28%	92,92%	82,55%
Generic		100	421	173	166	41,09%	95,95%	67,05%

Domains	noun recall	noun prec.	adj recall	adj prec.	verb recall	verb prec.	# tot nouns	# tot adj	# tot verbs
Tourism	64.52%	92.86%	28.72%	89.29%	9.18%	77.78%	868	195	294
Generic	58.27%	95.95%	28.38%	95.24%	5.32%	80%	254	74	94

Table 1a) Performance of the gloss disambiguation algorithm b) Performance by morphological category.

4.2 Evaluation of gloss disambiguation task

Ontology learning is perhaps the task for which our SSI algorithm is best suited: in fact, words in a complex terminological string are expected to be highly related, since the assumption is that the meaning of the entire string may be compositionally obtained from the individual components.
To test the SSI algorithm, we selected 650 complex terms from a set of 3,840 complex nominals extracted from a Tourism corpus and we manually assigned the appropriate WordNet synset to each word composing the term. We did the same for 200 terms in an Economy corpus (Wall Street Journal). Annotation is a rather subjective task, especially since in WordNet certain senses are very close.

We used two annotators with adjustment to reduce this problem. The results of the evaluation are shown in Figure 4. The light and dark towers represent, respectively, the precision and recall of the algorithm, for the Economy and Tourism domains (towers on the right side). Performance is measured also as a function of the number of complex terms from which the initial context *T* is built (e.g. Figure 2), indicated by |T| in the Figure (towers on the left side). The results show that precision is stable[12] around 80%, while the recall considerably increases, as expected, with |T|.

4.3 Evaluation of query expansion task

The objective of the query expansion experiment was to obtain a better insight on the *use* of sense information for improving web search. In a (still preliminary) study (described also in [19]) we showed that it is possible to obtain up to 22% systematic improvement in precision (over the first 10 retrieved pages using Google) when the

[12] due to the use of a threshold for sense selection

initial query is expanded with words related to a given word sense by contextual relations (namely, *gloss* and *topic*). The experiment has been conducted using the query topics of the TREC 2001[13] web track.

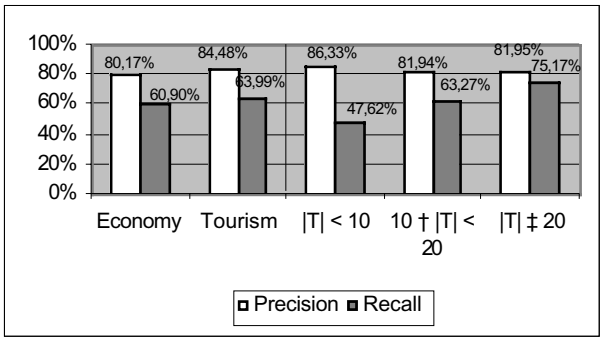

Figure 4. Summary evaluation of SSI algorithm on Economy and Tourism, by domain and by dimension of the initial context.

We experimented four sense-based expansion methods:

Synset expansion: "expansible" words are replaced by their synsets, retrieved by the algorithm of previous section.

Hyperonym expansion: "expansible" words are augmented by their WordNet direct hyperonyms.

Gloss synset expansion: "expansible" words are augmented by the synsets of its glosses (disambiguated by an ad-hoc version of our WSD algorithm).

Gloss-topic words expansion: "expansible" words are augmented with the words related to the concepts linked by *gloss* and *topic* relations.

Expansible words w_i are those for which the SSI algorithm assigns a sense with a confidence expressed by $f_I(S_{w_i}) > \beta$, where β is an experimentally tuned threshold and S is a sense of w_i. We queried the web with the first 24 of the 50 queries used in the TREC2001 web track. The queries (called "topics") include the actual query (*title*) but also text to explain the query (*description*) and describe precisely the type of documents that should be considered relevant (*narrative*). For example:

<TOP>
<num> Number: 518
<title> how we use statistics to aid our decision making?
<desc> Description:
Find documents that reference the use of statistical
data in decision-making.
<narr> Narrative:
A relevant document will describe a specific statistical method that is used to assist decision-making.
</top>

[13] TREC is a public challenge that runs every year on specific tasks, called tracks. The web site is http://trec.nist.gov/

Clearly, *description* and *narrative* texts cannot be used to expand the query, but only to manually verify the correctness of retrieved documents, as we did. To query the web, we used Google, which revealed not to be the best choice to exploit our algorithm, due to the limitation of 10 words per query. Therefore, for longer queries we are faced with the problem of choosing only a fragment of the candidate expansion words. However, we felt that our results could be stronger if they show an improvement in performance using the most popular search engine.

For each query, we retrieved the first 10 top ranked pages without query expansion, and then we repeated the search with each of the sense based expansion methods outlined above. When expansion terms are synsets, terms of a synset are put in OR. Whenever plain query terms + expansion terms exceed the threshold imposed by Google, we simply choose the first words of the list, a strategy that is certainly not optimized. The results are shown in Table 2 ($\beta \geq 0.8$).

Table 2 Retrieved correct pages when using sense information for all disambiguated words

Query short description	Plain Query words	+Synonyms	+Hyperonyms	+Gloss synsets	+Gloss/ Topic words
Avg. correct pages over first 10	5.125	5.291667	5.125	5.291667	6.291667
% variation with respect to plain query words		**+3.25%**	**+1.63%**	**+3.25%**	**+22.76%**

The scope of the experiment is rather limited and preliminary, and there is room for many improvements. For example, no additional mechanisms have been adopted to parse the query, e.g. named entity recognition, nor we conceived a more refined strategy to build a boolean query using sense information. Finally, more sophisticated expansion strategies could be used, exploiting intersecting patterns among sense graphs. Yet, Table 2 shows very interesting results: column 5 shows that the gloss/topic expansion strategy achieves a 23% improvement over the plain query words (column 1), and the improvement is consistent throughout the examples (in 19 over 24 queries). The experiment also shows, not surprisingly, that concepts occurring in the same semantic domain (e.g. *taxi* and *driver*) are best suited for query expansion than concepts related by taxonomic relations (e.g. *taxi* and *vehicle*).

4.4 Evaluation of a generic WSD task

In a final experiment, we evaluated the SSI algorithm on a standard WSD test set: the "all-words" test set of Senseval-2. In this sense evaluation task, participants were requested to disambiguate all the words in a generic text of 242 sentences, for a total of 2,473 words. For example:

<text id="d00">
The
<head id="d00.s00.t01">**art**</head>
of
<head id="d00.s00.t03">**change-ringing**</head>
<head id="d00.s00.t04">**is**</head>
<head id="d00.s00.t05">**peculiar**</head>
to
the
<head id="d00.s00.t08">**English**</head>

In the example, all the words annotated with a *head* tag must be disambiguated, including words with limited semantic content (e.g. **is**).
Best systems are those achieving both high precision and recall, an evaluation policy that perhaps is not best suited for many WSD applications (e.g. query expansion in previous section). Despite the Senseval task does not depicts a realistic application environment (though presumably this wasn't an objective for the organizers), we felt that an evaluation on a standard test set would be useful to obtain an unbiased comparison with other WSD methods in literature. For a fair comparison, we ran the experiment using the scoring software provided by the challenge organizers, available on the Senseval-2 web site. The result is shown in Figure 5.

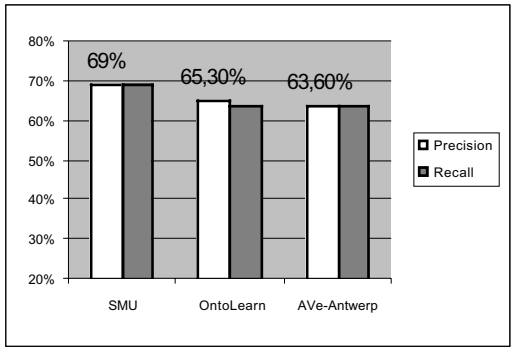

Figure 5. Precision and recall of the SSI algorithm compared with the first and second best systems in the Senseval-2 "all words" task.

The figure shows that SSI performs in between the first and second best systems in the Senseval-2 context, out of 26 participant systems. However, SSI is an untrained algorithm, while most Senseval-2 systems (including the first two) have been trained on available semantically tagged corpora. Notice that SSI *could* be trained, for example, through ad-hoc[14] tuning of the rule weights mentioned in Section 2.3. However, ad-hoc training would contradict the general-purpose nature of the SSI methodology.

Conclusions

SSI is an open research area in our group, and several improvements are being explored. The algorithm can be improved both through an enrichment of the structured representation of word senses, and through a refinement and extension of the grammar *G*. In the current version, grammar rules seek for patterns of conceptual relations (graph edges), but more complex rewriting rules could be defined, involving constraint specifications and type checking on concepts (graph nodes). Finally, machine-learning algorithms could be used to learn recurrent intersection patterns among word senses from semantically annotated corpora, as they are growingly made available from the computational linguistics community.

[14] "Ad-hoc" training implies learning from a semantically annotated corpus of word usage examples for at least some of the 2,473 words in the data set.

Acknowledgements

This work has been partly funded by the INTEROP network of excellence.

References

[1] R. Mihalcea, D. I. Moldovan: "A Highly Accurate Bootstrapping Algorithm for Word Sense Disambiguation" *International Journal on Artificial Intelligence Tools* 10(1-2): 5-21 (2001)
[2] Julio Gonzalo Felisa Verdejo Irina Chugur Juan Cigarr'an "Indexing with WordNet synsets can improve text retrieval" *Proceedings of the COLING/ACL '98 Workshop on Usage of WordNet for NLP* (1998)
[3] Tim Berners-Lee "Semantic Web Roadmap" http://www.w3.org/DesignIssues/Semantic.html
[4] World Wide Web Consortium "W3C A to Z" http://www.w3.org/
[5] OpenCyc home page http://www.opencyc.org/
[6] WordNet home page http://www.cogsci.princeton.edu/~wn/
[7] FrameNet home page http://www.icsi.berkeley.edu/~framenet/
[8] K.S. Fu *Syntactic Pattern Recognition and Applications* Prentice-Hall, Engelwood Cliffs, NJ, 1982
[9] H. Bunke and A. Sanfeliu (editors) *Syntactic and Structural pattern Recognition: Theory and Applications* World Scientific, Series in CS vol. 7, 1990.
[10] N. Guarino and C. Welty "Evaluating Ontological Decisions with OntoClean ", Communications of the ACM 45-(2) (61-65) 2002.
[11] *2nd Global WordNet Conference*, Brno, Czech Republic, January 20-23, 2004, http://www.fi.muni.cz/gwc2004/
[12] B. Magnini and G. Cavaglia "Integrating Subject Field Codes into WordNet", *Proceedings of the 2nd* LREC2000, Atenas 2000.
[13] A. Gangemi, R. Navigli and P. Velardi "The OntoWordNet Project: extension and axiomatization of conceptual relations in WordNet", *2nd. International Conference on Ontologies, Databases and Applications of Semantics* (ODBASE 2003), ed. Springer Verlag, 3-7 November 2003, Catania, Sicily (Italy).
[14] R. Basili, M.T. Pazienza, and P. Velardi, "Integrating general purpose and corpus-based verb classification", *Computational Linguistics*, MIT press, December 1996.
[15] M. Missikoff, R. Navigli and P. Velardi "Integrated Approach for Web Ontology Learning and Engineering", *IEEE Computer*, November 2002.
[16] R. Navigli and P. Velardi "Learning Domain Ontologies from Document Warehouses and Dedicated Web Sites", *Computational Linguistics*, MIT press, in press, 2004.
[17] E. Voorhees "Using WordNet to disambiguate Word Senses for Text retrieval", ACM-SIGIR, Pittsburgh, PA, 1993.
[18] M. Sanderson "Word Sense Disambiguation and Information Retrieval", 17th Int. Conf. on Research and Development in Information Retrieval, 1994.
[19] R. Navigli and P. Velardi "An analysis of ontology-based query expansion strategies" in *Workshop on Adaptive Text Extraction and Mining*, Monday 22 September 2003 Cavtat-Dubrovnik (Croatia), held in conjunction with 14th European Conference on Machine Learning (ECML)..