Structural Semantic Interconnections: A Knowledge-Based Approach to Word Sense Disambiguation

Roberto Navigli and Paola Velardi

Abstract—Word Sense Disambiguation (WSD) is traditionally considered an AI-hard problem. A break-through in this field would have a significant impact on many relevant Web-based applications, such as Web information retrieval, improved access to Web services, information extraction, etc. Early approaches to WSD, based on knowledge representation techniques, have been replaced in the past few years by more robust machine learning and statistical techniques. The results of recent comparative evaluations of WSD systems, however, show that these methods have inherent limitations. On the other hand, the increasing availability of large-scale, rich lexical knowledge resources seems to provide new challenges to knowledge-based approaches. In this paper, we present a method, called *structural semantic interconnections* (*SSI*), which 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. Sense specifications are created from several available lexical resources that we integrated in part manually, in part with the help of automatic procedures. The SSI algorithm has been applied to different semantic disambiguation problems, like automatic ontology population, disambiguation of sentences in generic texts, disambiguation of words in glossary definitions. Evaluation experiments have been performed on specific knowledge domains (e.g., tourism, computer networks, enterprise interoperability), as well as on standard disambiguation test sets.

Index Terms—Natural language processing, ontology learning, structural pattern matching, word sense disambiguation.

---

1 INTRODUCTION

WORD sense disambiguation (WSD) is perhaps the most critical task in the area of computational linguistics (see [1] for a survey). Early approaches were based on semantic knowledge that was either manually encoded [2], [3] or automatically extracted from existing lexical resources, such as WordNet [4], [5], LDOCE [6], and Roget's thesaurus [7]. Similarly to other artificial intelligence applications, knowledge-based WSD was faced with the knowledge acquisition bottleneck. Manual acquisition is a heavy and endless task, while online dictionaries provide semantic information in a mostly unstructured way, making it difficult for a computer program to exploit the encoded lexical knowledge.

More recently, the use of machine learning, statistical and algebraic methods ([8], [9]) prevailed on knowledge-based methods, a tendency that clearly emerges in the main Information Retrieval conferences and in comparative system evaluations, such as SIGIR,[1] TREC,[2] and SensEval.[3] These methods are often based on training data (mainly, word cooccurrences) extracted from document archives and from the Web.

The SensEval workshop series are specifically dedicated to the evaluation of WSD algorithms. Systems compete on different tasks (e.g., full WSD on generic texts, disambiguation of dictionary sense definitions, automatic labeling of semantic roles) and in different languages. *English All-Words* (full WSD on annotated corpora, such as the Wall Street Journal and the Brown Corpus) is among the most attended competitions. At Senseval-3, held in March 2004, 17 supervised and 9 unsupervised systems participated in the task. The best systems were those using a combination of several machine learning methods, trained with data on word cooccurrences and, in few cases, with syntactic features, but nearly no system used semantic information.[4] The best systems reached about 65 percent precision, 65 percent recall,[5] a performance considered well below the needs of many real-world applications [10]. Comparing performances and trends with respect to previous SensEval events, the feeling is that supervised machine learning methods have little hope of providing a real break-through, the major problem being the need for high quality training data for all the words to be disambiguated.

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. On the other hand, new emerging applications, like the so-called Semantic Web [11], foster "an extension of the current web in which information is given well-defined *meaning*,

---

1. http://www.acm.org/sigir/.
2. http://trec.nist.gov/.
3. http://www.senseval.org/.

● *The authors are with the Dipartimento di Informatica, Università of Roma "La Sapienza," via Salaria 113, 00198 Roma, Italy.
E-mail: {navigli, velardi}@di.uniroma.it.*

4. One of the systems reported the use of domain labels, e.g., medicine, tourism, etc.
5. A performance sensibly lower than for Senseval-2.

better enabling computers and people to work in cooperation," an objective that could hardly be met by manual semantic annotations. Large-scale semantic annotation projects would greatly benefit from robust methods for automatic sense selection.

In recent years, the results of many research efforts for the construction of online lexical knowledge repositories, ontologies and glossaries became available (e.g., [12], [13], [14]), creating new opportunities for knowledge-based sense disambiguation methods. The problem is that these resources are often heterogeneous,[6] midway formal, and sometimes inconsistent. Despite these problems, we believe that the future of semantic annotation methods critically depends on the outcomes of large-scale efforts to integrate existing lexical resources and on the design of WSD algorithms that exploit this knowledge at best.

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, collocation inventories, annotated corpora, and glossaries, that we combined in part manually, in part automatically.

The paper is organized as follows: In Section 2, we describe the procedure for creating structured (graph) representations of word senses from a variety of lexical resources. Section 3 presents the structural semantic interconnection algorithm and describes the context-free grammar for detecting semantic interconnections. Section 4 provides implementation details for three word sense disambiguation problems. Finally, Section 5 is dedicated to the description of several experiments that we made on standard and domain-specific testing environments. The latter refer to past and on-going national and European projects in which we participate.

2   CREATING A GRAPH REPRESENTATION OF WORD SENSES

Our approach to word sense disambiguation lies in the *structural pattern recognition* framework. Structural or syntactic pattern recognition [15], [16] 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 classification performances. 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.

Word senses clearly fall under the category of objects that are better described through a set of structured features.

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, large lexical knowledge bases and annotated resources offer an ideal starting point for constructing structured representations of word senses. In these repositories, lexical

knowledge is described with a variable degree of formality and many criticisms of the consistency and soundness (with reference to computer science standards) of the encoded information have been made. Despite these criticisms and efforts to overcome some limitations (e.g., the OntoClean project [17]), these knowledge repositories became highly popular to the point where dedicated conferences are organized each year among the scientists that use these resources for a variety of applications in the information technology area (e.g., [18] and others).

2.1   Online Lexical Knowledge Repositories

Graph representations of word senses are automatically generated from a lexical knowledge base (LKB) that we built integrating a variety of online resources. This section describes the resources that we used and the integration procedure.

1. <u>WordNet 2.0</u>[7] [13], a *lexicalized online ontology* including over 110,000 concepts. WordNet is a lexical ontology in that concepts correspond to word senses. Concept names (or concept labels) are called *synsets*. Synsets are groups of synonym words that are meant to suggest an unambiguous meaning, e.g., for bus#1: "*bus, autobus, coach, charabanc, double-decker, jitney, motorbus, motor coach, omnibus.*"[8] In addition to synsets, the following information is provided for word senses:

    i.   a textual sense definition called *gloss* (e.g., *coach*#5: "a vehicle carrying many passengers; used for public transport");

    ii. hyperonymy-hyponymy links (i.e., kind-of relations, e.g., *bus*#1 is a kind of *public-transport*#1);

    iii. meronymy-holonymy relations (i.e., part-of relations, e.g., *bus*#1 has part *roof*#2 and *window*#2);

    iv. other syntactic-semantic relations, as detailed later, some of which are not systematically provided.

2. <u>Domain labels</u> described in [19]. This resource assigns a domain label (e.g., *tourism, zoology, sport,* etc.) to most WordNet synsets.[9]

3. <u>Annotated corpora.</u> Texts provide examples of *word sense usages in context*. From these texts, we automatically extract cooccurrence information. Co-occurrences are lists of words that co-occur in the same context (usually a sentence). A *semantic co-occurrence* is a list of co-occurring word *senses* or concepts.

    We extract co-occurrences from the following resources:

    a.   *SemCor* [20] is a corpus where each word in a sentence is assigned a sense selected from the WordNet sense inventory for that word;[10] an example of a SemCor document is the following:

    ***Color*#1** *was **delayed*#1** *until 1935, the **widescreen*#1** *until the **early*#1 **fifties*#1**.

---

7. WordNet is available at http://wordnet.princeton.edu.

8. In what follows, WordNet synsets are indicated both by numbered words (e.g., *chief*#1) and by the list of synonyms (e.g., bus, autobus, coach, etc.). Synsets identify a concept or word sense unambiguously.

9. WordNet 2.0 also provides domain labels. However, we preferred the lable data set described in [19].

10. Semcor: http://www.cs.unt.edu/~rada/downloads.html#semcor.

6. That is, they use different inventories of semantic realtions, different concept names, different strategies for representing roles, properties, and abstract categories, etc.

From this sentence, the following semantic co-occurrence is generated: (color#1 delay#1 wide-screen#1 early#1 fifties#1).

b. _LDC-DSO_ [21] is a corpus where each document is a collection of sentences having a certain word in common.[11] The corpus provides a sense tag for each occurrence of the word within the document. Examples from the document focused on the noun _house_ are the following:

*Ten years ago, he had come to the **house#2** to be interviewed.*

*Halfway across the **house#1**, he could have smelled her morning perfume.*

The cooccurrence generated from the second sentence is: (house#1 smell morning perfume), where only the first element in the list is a word sense.

c. _WordNet glosses and WordNet usage examples._ In WordNet, besides textual definitions, _examples_ referring to synsets rather than to words are sometimes provided. From these examples, as well as from glosses, a co-occurrence list can be extracted. Some examples follow:

*"Overnight **accommodations#4** are available."*
*"Is there **intelligent#1** life in the universe?"*
*"An **intelligent#1** question."*

As for the LDC corpus, these co-occurrences are mixed, i.e., they include both words and word senses.

4. Dictionaries of collocations, i.e., _Oxford Collocations_ [22], _Longman Language Activator_ [23], and _Lexical FreeNet_.[12] Collocations are lists of words that belong to a given semantic domain, e.g., (_bus, stop, station_) and (_bus, network, communication_). More precisely, collocations are words that cooccur with a mutual expectancy greater than chance. The members of a list are called _collocates_.

Even co-occurrences extracted from corpora, as in item 3, subsume a common semantic domain, but, since they are extracted from running texts without statistical or manual filters, they only approximate the more reliable information provided by dictionaries of collocations.

Online dictionaries provide collocations for different senses of each word, but there are no explicit pointers between dictionary word senses and WordNet synsets, contrary to the resources described in items 2 and 3. In order to integrate collocations with our lexical knowledge base, we proceeded as follows (more details are found in [24]):

First, we identified a set of _representative words_ (RW), precisely, the restricted vocabulary used by _Longman_ to write definitions. These RW (about 2,000) correspond to about 10,000 WordNet synsets, that we call _representative concepts_ (RC). RCs are shown in [24] to provide a good semantic coverage of the entire WordNet.

Second, we manually associated the collocations extracted from the aforementioned dictionaries to a fragment of these representative concepts (863 RCs, corresponding to 248 RWs). After this step, we obtain collocations relating a representative _concept_ (identified by a WordNet synset) with a list of possibly ambiguous _words_. With reference to the previous

examples of collocates, we obtain (coach#5 bus taxi) and (coach#1 football sport). Notice again that mixed concept-word associations are also extracted from LDC-DSO, from WordNet glosses, and usage examples, while, in SemCor, all the elements in a co-occurrence list are word senses.

Third, in order to obtain fully _semantic_ associations, i.e., lists of related senses, we apply the WSD algorithm described in the rest of this paper to these associations. The algorithm takes an already disambiguated word (e.g., $bus\#1$) as input and attempts to disambiguate all its associated words (e.g., _stop, station_) with respect to the available synset. This disambiguation step is also applied to mixed cooccurrences extracted from corpora.

This disambiguation experiment is described in Sections 4 and 5.

2.2 Building Graph Representations for Word Senses

As detailed in the previous section, we created a lexical knowledge base (LKB) including semantic relations explicitly encoded in WordNet and semantic relations extracted from annotated corpora and dictionaries of collocations.

The LKB is used to generate labeled directed graph (_digraph_) representations of word senses. We call these _semantic graphs_ since they represent alternative conceptualizations for a given lexical item.

Fig. 1 shows an example of the semantic graphs generated for senses #1 (_vehicle_) and #2 (_connector_) of bus, 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 Fig. 1. This distance has been experimentally tuned to optimize WSD performance.

The following set of semantic relations is included in a semantic graph: _hyperonymy_ (car#1 _is a kind of_ vehicle#1, denoted with $\to^{\text{kind-of}}$), _hyponymy_ (its inverse, $\to^{\text{has-kind}}$), _meronymy_ (room#1 _has-part_ wall#1, $\to^{\text{has-part}}$), _holonymy_ (its inverse, $\to^{\text{part-of}}$), _pertainymy_ (dental#1 _pertains-to_ tooth#1 $\to^{\text{pert}}$), _attribute_ (dry#1 _value-of_ wetness#1, $\to^{\text{attr}}$), _similarity_ (beautiful#1 _similar-to_ pretty#1, $\to^{\text{sim}}$), _gloss_ ($\to^{\text{gloss}}$), _context_ ($\to^{\text{context}}$), _domain_ ($\to^{\text{dl}}$). All these relations are explicitly encoded in WordNet, except for the latter three.

Context expresses semantic associations between concepts ($food\#2$ _has context_ $drink\#3$), extracted from annotated corpora, usage examples, and collocation dictionaries, as explained in Section 2.1. _Gloss_ relates a concept with another concept occurring in its WordNet natural language definition ($coach\#5$ _has in gloss_ $passenger\#1$). Finally, _domain_ relates a concept with its domain label ($terminal\#3$ _has domain computer science_).

3 THE STRUCTURAL SEMANTIC INTERCONNECTION ALGORITHM

This section presents the Structural Semantic Interconnection algorithm (SSI), a knowledge-based iterative approach to Word Sense Disambiguation. For the sake of generality, we do not refer here to WordNet word senses, synsets, and LKB, but to concepts, concept labels, and ontology.

The classification problem can be stated as follows:

- T (the _lexical context_) is a list of related terms.
- t is a term in T to be disambiguated.

11. LDC-DSO: http//www.ldc.upenn.edu/.
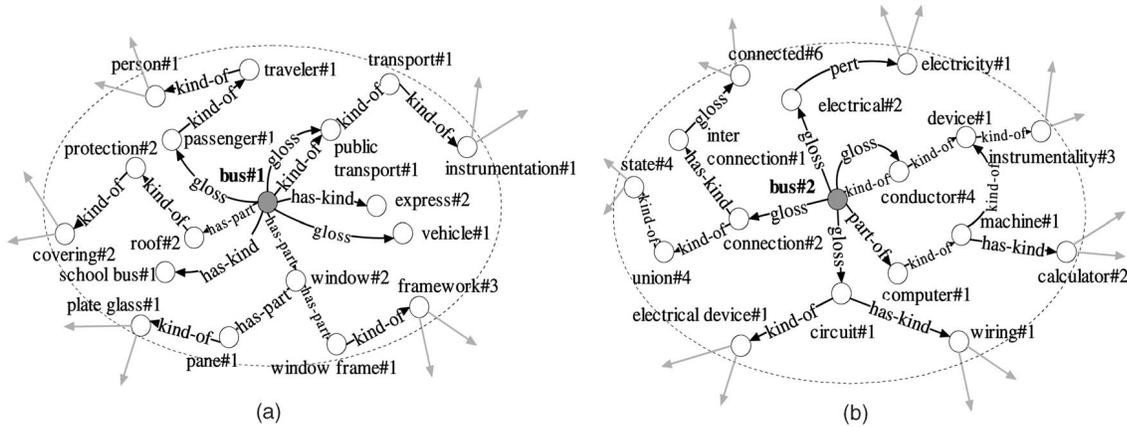12. Lexical FreeNet: http://www.lexfn.com.

Fig. 1. Graph representations for (a) sense #1 and (b) sense #2 of bus.

- $S_1^t, S_2^t, \ldots, S_n^t$, are structural specifications of the possible concepts for t (precisely, *semantic graphs*).
- I (the *semantic context*) is a list of structural specifications of the concepts associated to (some of) the terms in T. I includes either one or no specifications for each term in $T \setminus \{t\}$ and no specification for t.
- G is a grammar describing relevant relations between structural specifications (precisely, *semantic interconnections* among graphs).
- Determine how well the structural specifications in I match that of each of $S_1^t, S_2^t, \ldots, S_n^t$, using G.
- Select the best matching S_i^t.

Structural specifications are built from available conceptualizations for the lexical items in T. We refer to such conceptualizations as to an ontology O.[13]

In the next section, we provide an overview of the algorithm, then, in subsequent sections, we will add implementation details. A complete example of an execution of the algorithm is illustrated in Section 4.

3.1 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 cooccurring terms $T = [t_1, \ldots, t_n]$ and a list of associated senses $I = [S^{t_1}, \ldots, S^{t_n}]$, i.e., the semantic interpretation of T, where S^{t_i} [14] is either the chosen sense for t_i (i.e., the result of a previous disambiguation step) or the null element (i.e., the term is not yet disambiguated).

A set of *pending* terms is also maintained, $P = \{t_i | S^{t_i} = \text{null}\}$. 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.

13. No assumption is made by the SSI algorithm about the semantic knowledge base used to build graphs except for the existence of a set of concepts related by semantic relations. Hence, we simply refer to this resource as to an ontology O.

14. Note that, with S^{t_i}, we refer interchangeably to the semantic graph associated with a sense or to the sense name.

Initially, the list I includes the senses of monosemous terms in T, or a fixed word sense. For example, if we have a representative concept S (see Section 2.1) and the list of its collocates, I initially includes the concept S (e.g., $bus\#1$) and P the collocates to be disambiguated (e.g., *stop, station*). Another case where an initial sense is available is when the task is disambiguating words in the definition (gloss) of a word sense: Then, I includes S (e.g., $bus\#1$) and P, the words in the gloss (e.g., *vehicle, carry, passenger, public transport*).

If no monosemous terms are found or if no initial synsets are provided, the algorithm makes an initial guess based on the most probable sense[15] of the less ambiguous term t. Then, the process is forked into as many executions as the total number of senses for t, as detailed later.

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 $f_I : C \times T \to \Re$, where T is the set of terms to be disambiguated and C is the set of all the concepts in the ontology O, defined as follows:

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

where $Senses(t)$ is the subset of concepts C in O associated with the term t and

$$\varphi(S, S') = \rho'(\{w(e_1 \cdot e_2 \cdot \ldots \cdot e_n) | \\ S \xrightarrow{e_1} S_1 \xrightarrow{e_2} \cdots \xrightarrow{e_{n-1}} S_{n-1} \xrightarrow{e_n} S'\}),$$

i.e., a function (ρ') of the weights (w) of each path connecting S with S', where S and S' are semantic graphs. A semantic path between two senses S and S', $S \xrightarrow{e_1} S_1 \xrightarrow{e_2} \cdots \xrightarrow{e_{n-1}} S_{n-1} \xrightarrow{e_n} S'$, 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 nonterminal symbols (N) encode

15. Probability estimates are often available. WordNet orders word senses by frequency of use.

(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 $A \rightarrow \alpha$ in P_G, where $A \in N$ and $\alpha \in (N \cup E)^*$, i.e., α is a sequence of terminal and nonterminal 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 $S^t = \arg\max_{S \in C} f_I(S, t)$ as the most likely interpretation of t and updates the list I with the chosen concept. A threshold can be applied to $f_I(S, t)$ to improve the robustness of system's choices.

At the end of a generic iteration, a number of terms are 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 $f_I(S, t) > 0$, 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.[16]

A special case of input for the SSI algorithm is given by $I = [-, -, \ldots, -]$,[17] that is, when no initial semantic context is available (there are no monosemous words in T). In this case, an initialization policy selects a term $t \in T$ and the execution is forked into as many processes as the number of senses of t. Let n be such a number. For each process i ($i = 1, \ldots, n$), the input is given by $I_i = [-, -, \ldots, S_i^t, \ldots, -]$, where S_i^t is the ith sense of t in $Senses(t)$. Each execution outputs a (partial or complete) semantic context I_i. Finally, the most likely context I_m is given by:

$$m = \underset{1 \leq i \leq n}{\arg\max} \sum_{S^{t_j} \in I_i} f_{I_i}(S^{t_j}, t_j).$$

A pseudocode of the SSI algorithm is reported in [27].

3.2 The Grammar

The grammar G has the purpose of describing meaningful interconnecting patterns among semantic graphs representing conceptualizations 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 labeled with e_i and e_{i+1} are incoming and/or outgoing from the same concept node, that is, $\overset{e_i}{\rightarrow}(S)\overset{e_{i+1}}{\rightarrow}, \overset{e_i}{\leftarrow}(S)\overset{e_{i+1}}{\rightarrow}, \overset{e_i}{\rightarrow}(S)\overset{e_{i+1}}{\leftarrow}, \overset{e_i}{\leftarrow}(S)\overset{e_{i+1}}{\rightarrow}$. 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 manually defined by inspecting the intersecting patterns automatically extracted from pairs of manually disambiguated word senses cooccurring in different domains. Some of the rules in G are inspired by previous work on the eXtended WordNet project described in [25]. The terminal symbols e_i are the conceptual

---

16. The SSI algorithm in its current version is a *greedy* algorithm. In each step, the "best" senses are chosen according to the current I and P, therefore, the order in which senses are chosen may affect the final result. An exhaustive search will be computationally feasible in a forthcoming optimized release.

17. We indicate a null element with the symbol -.

$$
\begin{array}{ll}
S_G \rightarrow S_s \mid S_g & \textit{(all the heuristics)} \\
S_s \rightarrow S_1 \mid S_2 \mid S_3 & \textit{(simple heuristics)} \\
S_1 \rightarrow E_1 S_1 \mid E_1 & \textit{(hyperonymy/meronymy)} \\
E_1 \rightarrow e_{\text{kind-of}} \mid e_{\text{part-of}} & \\
S_2 \rightarrow E_2 S_2 \mid E_2 & \textit{(hyponymy/holonymy)} \\
E_2 \rightarrow e_{\text{has-kind}} \mid e_{\text{has-part}} & \\
S_3 \rightarrow e_{\text{kind-of}} S_3 e_{\text{has-kind}} \mid e_{\text{kind-of}} e_{\text{has-kind}} & \textit{(parallelism)} \\
S_g \rightarrow e_{\text{gloss}} S_s \mid S_4 \mid S_5 & \textit{(gloss)} \\
S_4 \rightarrow e_{\text{gloss}} \mid e_{\text{topic}} & \textit{(gloss, context)} \\
S_5 \rightarrow e_{\text{gloss}} e_{\text{is-in-gloss}} & \textit{(gloss+gloss}^{-1}\textit{)}
\end{array}
$$

Fig. 2. An excerpt of the context-free grammar for the recognition of semantic interconnections.

relations extracted from WordNet and other online lexical-semantic resources, as described in Section 2.

G is defined as a quadruple (E, N, S_G, P_G), where

$$E = \{e_{\text{kind-of}}, e_{\text{has-kind}}, e_{\text{part-of}}, e_{\text{has-part}}, e_{\text{gloss}}, e_{\text{is-in-gloss}}, e_{\text{topic}}, \ldots\},$$

$N = \{S_G, S_s, S_g, S_1, S_2, S_3, S_4, S_5, E_1, E_2, \ldots\}$, and P_G includes about 40 productions. An excerpt of the grammar is shown in Fig. 2.

As stated in the 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 $S_G \Rightarrow^+ e_1 \cdot e_2 \cdot \ldots \cdot e_n$. The weights of patterns are automatically learned using a perceptron model [26]. The weight function is given by:

$$weight(pattern_j) = \alpha_j + \beta_j \left(\frac{1}{length\_pattern_j} \right),$$

where α_j is the weight of rule j in G and the second addend is a smoothing parameter inversely proportional to the length of the matching pattern. The perceptron has been trained on the SemCor semantically annotated corpus (see Section 2).

Two examples of rules with a high α_j are the *hyperonymy/meronymy rule* and the *parallelism rule*. In the first rule, two concepts are related by a sequence of hyperonymy/meronymy relations, e.g.:

$$mountain\#1 \overset{has\text{-}part}{\longrightarrow} mountain\_peak\#1 \overset{kind\text{-}of}{\longrightarrow} top\#3.$$

The *parallelism rule* connects two concepts having a common ancestor, e.g.:

$$organization\#1 \overset{kind\text{-}of}{\longrightarrow} enterprise\#2 \overset{kind\text{-}of}{\longleftarrow} company\#1.$$

Detailed comments on the rules in G are found in [27]. More examples of patterns matching the rules in G are provided in Section 4.

4 THREE APPLICATIONS OF THE SSI ALGORITHM

The SSI algorithm has been applied in several different WSD tasks. Three of these applications are discussed in this section. Section 5 provides an evaluation for each of these tasks, as well as an evaluation on a "standard" WSD disambiguation task, where all the words in a sentence must be disambiguated.

4.1 Disambiguation of Textual Definitions in an Ontology or Glossary

Glossaries, ontologies, and *thesauri* 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 WordNet, the word *enterprise* has three senses and *material* has six. Associating the correct sense with each word in a gloss is a sort of preliminary step to construct formal concept definitions from informal ones.

For the gloss disambiguation task, the SSI algorithm is initialized as follows: In Step 1, the list I includes the sense S whose gloss we wish to disambiguate and the list P includes all the terms in the gloss <u>and</u> in the gloss of the hyperonym of S. Words in the hyperonym's gloss are useful to extend the context available for disambiguation. As shown in Section 5, large contexts have a positive influence on the performance of SSI.

In the following, we present a sample execution of the SSI algorithm for the gloss disambiguation task applied to the WordNet 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:[18]

$$T = [retrospective, work, object, exhibition, life, statue,$$
$$artist, selection, representative, painting, art]$$
$$I = [retrospective\#1, -, -, -, -, -, -, -, -, -, -]$$
$$P = [work, object, exhibition, life, statue, artist, selection,$$
$$representative, painting, art].$$

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 *hyponymy/meronymy* paths:[19]

$$retrospective\#1 \overset{kind\text{-}of^2}{\longrightarrow} exhibition\#2,$$
$$statue\#1 \overset{kind\text{-}of^3}{\longrightarrow} art\#1 \text{ and } statue\#1 \overset{kind\text{-}of^6}{\longrightarrow} object\#1.$$

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 is found:

$$art\#1 \overset{has\text{-}kind^2}{\longrightarrow} painting\#1 \text{ (painting is a kind of art),}$$

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 path (*gloss/context* rule) between $artist\#1$ and sense 12 of *life* (*biography, life history*): $artist\#1 \overset{context}{\longrightarrow} life\#12$. 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: $retrospective\#1$ = "An $art\#1$ $exhibition\#2$ of a representative selection of an $artist\#1$'s $life\#12$ work."

4.2 Disambiguation of Word Collocations

The second application has been already outlined in Section 2.1, item 4. The objective is to obtain fully semantic associations from lists of collocations where only one word has been disambiguated, the so-called *representative word*. The algorithm is applied in a way that is similar to the case of gloss disambiguation.

The initial context is $T = [w, t_1, t_2, \ldots, t_n]$, where w is the representative word (see Section 2.1). I is initialized as $[S^w, -, -, \ldots, -]$, where S^w is the representative concept corresponding to w.

As an example, consider the representative concept $house\#1$, exposing, among other things, the following collocations: *apartment, room, wall, floor, window, guest, wing*.

The initial context T is given by [<u>house</u>, *apartment, room, wall, floor, window, guest, wing*], while I after the first step is: $[house\#1, apartment\#1, -, -, -, -, -, -]$ (*apartment* is monosemous). The final outcome of SSI is the set

$$I = [house\#1, apartment\#1, room\#1, wall\#1, floor\#1,$$
$$window\#1, guest\#1, wing\#9],$$

where all words have been disambiguated. The semantic patterns identified by SSI are illustrated in Fig. 3.

4.3 Automatic Ontology Learning

The SSI algorithm is the core of a domain-ontology learning system, OntoLearn, used to create trimmed and domain-tuned views of WordNet.

OntoLearn uses evidence extracted from glossaries and document repositories, usually available in a given domain or Web community, to build a *forest of domain concepts*, which are then used to extend an already existing ontology.

The following steps are performed by the system:[20]

4.3.1 Extract Pertinent Domain Terminology

Simple and multiword expressions are automatically extracted from domain-related corpora, like enterprise interoperability (e.g., *collaborative work*), tourism (e.g., *room*

---

18. From now on, we omit null elements.

19. With $S \overset{R\ \ i}{\longrightarrow} S'$, we denote a path of i consecutive edges labeled with the relation R interconnecting S with S'.

20. Limited details on the algorithms are provided here for the obvious sake of space. The interested reader can access the referenced OntoLearn bibliography, especially [27].
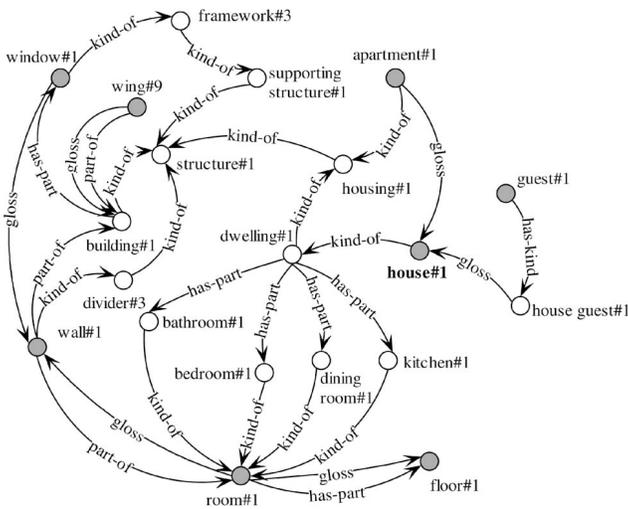
Fig. 3. Semantic patterns connecting house#1 with related concepts.

reservation), computer network (e.g., *packet switching network*), art techniques (e.g., *chiaroscuro*). Statistical and natural language processing (NLP) tools are used for automatic extraction of terms [27].

Statistical techniques are specifically aimed at simulating *human consensus* in accepting new domain terms. Only terms uniquely and consistently found in domain-related documents <u>and</u> not found in other domains used for contrast are selected as candidates for the domain terminology.

The performance of this task critically depends upon the availability of domain-relevant documentation, usually provided by domain experts.

4.3.2 Web Search of Available Natural Language Definitions from Glossaries or Documents

Available natural language definitions are searched on the Web using online glossaries or extracting "*definitory*" sentences in available documents. A context-free (CF) grammar is used to extract definitions. An excerpt is shown in Fig. 4.

In this example, S, NP, and PP stand for sentence, noun phrase, and prepositional phrase, respectively. KIND1 captures the portion of the sentence that identifies the hyperonym in the definition.

This grammar fragment identifies (and analyzses) definitory sentences such as: "[In a programming language]$_{PP}$, [an *aggregate*]$_{NP}$ [that consists of data objects with identical attributes, each of which may be uniquely referenced by subscription]$_{SEP}$," which is a definition of *array* in a computer network domain.

The grammar is tuned for high precision, low recall. In fact, certain expressions (e.g., *X is a Y*) are overly general and produce mostly noise when used for searching definitions in free texts.

4.3.3 Filter Out Nonrelevant Definitions

Multiple definitions may be found when searching in glossaries on the Internet. Some may be not pertinent to the selected domain (e.g., in the interoperability domain, *federation* as "the forming of a nation" is not pertinent, while a more appropriate definition is "a common object model, supporting Runtime Infrastructure"). A statistical filtering algorithm is used to prune out "noisy" definitions, based on a probabilistic model of the domain.

```
S → PP ',' NP SEP
NP → N1 KIND1
KIND1 → MOD1 NOUN1
MOD1 → Verb | Adj | Verb ',' MOD1 | Adj ',' MOD1
NOUN1 → Noun
N1 → Art | Adj
SEP → ',' | '.' | Prep | Verb | Wh
PP → Prep NP
```
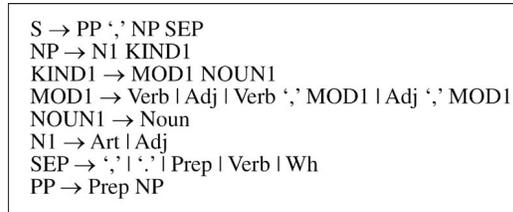
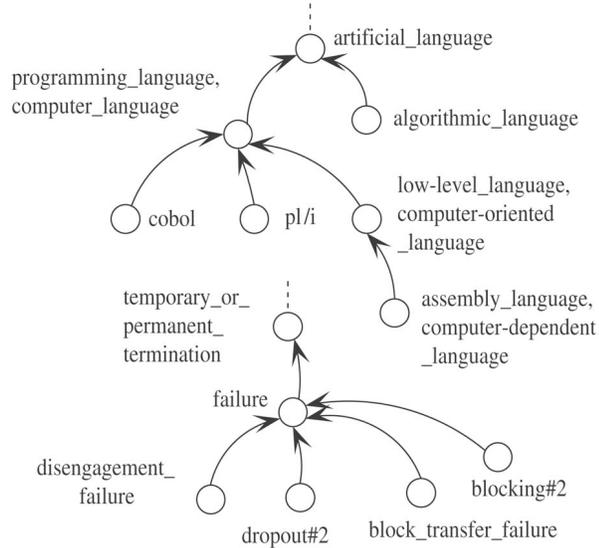Fig. 4. An excerpt of a grammar to extract and parse definitions.

Fig. 5. Examples of taxonomic trees of terms (from the computer network domain).

4.3.4 Parse Definitions to Extract Kind-of Information

The CF grammar of Fig. 4 is again used to extract hyperonymy relations from natural language definitions. For example, in the *array* example reported above, the same grammar rule shown in Fig. 4 can be used to extract the information (corresponding to the KIND1 segment in the grammar excerpt): $array \xrightarrow{kind\text{-}of} aggregate$.

If definitions are not found in available resources, alternative approaches are used, e.g., creating a definition compositionally from its constituents. This is not discussed here since it is outside the scope of the paper.

4.3.5 Arrange Terms in Hierarchical Trees

Terms are arranged in forests of trees, according to the information extracted in Section 4.3.4. Fig. 5 shows examples from a computer network domain.

4.3.6 Link Subhierarchies to the Concepts of a Core Ontology

The semantic disambiguation algorithm SSI is used to append subtrees under the appropriate node of a generic core ontology. In our work, we use WordNet, but other generic ontologies can be employed, if available. The preference for WordNet is motivated by the fact that sufficiently rich domain ontologies are currently available only in a few domains (e.g., medicine). For small core ontologies (i.e., CRM-CIDOC[21] in the field of cultural heritage), it is relatively

---

21. http://cidoc.ics.forth.gr/index.html. CRM-CIDOC has on the order of 50 concepts.

easy to manually draw links between core concepts and the corresponding WordNet synsets.

With reference to Fig. 5, the root *artificial language* has a monosemous correspondent in WordNet, but *temporary or permanent termination* has no direct correspondent. The node is then linked to *termination*, but, first, a disambiguation problem must be solved since *termination* in WordNet has two senses: "end of a time span" and "expiration of a contract"; therefore disambiguation is necessary.

For this WSD task, the SSI algorithm works as follows: *I* is initialized with one of the alternative concepts to which the subtree is to be related (e.g., $I = [termination\#1]$). *P* is the list of multiword expressions, or components of these expressions, that appear in the subtree <u>and</u> have a lexical correspondent in WordNet. For example, from the tree rooted in *temporary-or-permanent-termination*, the following list *P* is generated (see the corresponding tree in Fig. 5):

$$P = \{disengagement, failure, block, transfer, dropout,$$
$$temporary, permanent\}.$$

The algorithm then forks into as many executions as the number of alternatives for the tree attachment (in our example, there are two alternatives, $termination\#1$ and $termination\#2$).

4.3.7 Provide the Output to Domain Specialists for Evaluation and Refinement

The outcome of the ontology learning process (a trimmed and extended version of WordNet) is then submitted to experts for corrections, extensions, and refinement.

In the current version of OntoLearn, the output of the system is a taxonomy, not an ontology, since the only information provided is the *kind-of* relation. However, extensions are in progress, aimed at extracting other types of relations from definitions and online lexical resources.

5 EXPERIMENTAL RESULTS

This section provides an evaluation of all the tasks described in the previous section, with the addition of a "standard" WSD task in which all the words in a generic sentence must be disambiguated.

5.1 Evaluation on the Senseval-3 Gloss Disambiguation Task

SSI participated in the Senseval-3 gloss disambiguation task, held in March 2004. The task [28] was based on the availability of disambiguated hand-tagged glosses (called *golden glosses*) created in the eXtended WordNet [25] project.

WordNet senses have been assigned to 42,491 content words, including adjectives, verbs, nouns, and adverbs. Participants were provided with all glosses from WordNet in which at least one word had a "gold" quality sense assignment. Systems were requested to disambiguate the highest number of such words, with the highest precision, i.e., both recall and precision had to be optimized. Evaluation was produced using a scoring program provided by the organizers.

The SSI algorithm in its "standard" version attempts to optimize only precision: If no semantic connections are found or if the weight of a connection is below a threshold, no sense choice is provided. We believe that this is a reasonable setting

for real-domain applications. We also demonstrated, in a query expansion application [29], that significant improvements in performance might be obtained if even a small fragment of the query words are disambiguated with at least 80 percent precision. In ontology learning, where the objective is to speed up the human task of ontology construction, it is far more productive to submit reliable data, even at the price of limited coverage.

For the purpose of the gloss competition, however, we submitted a version of the system in which SSI was forced to produce a sense choice for all the words to be disambiguated. The threshold on pattern weights was removed and the first WordNet sense was selected every time since no interconnection patterns could be found for any of the senses of a word.[22]

Finally, at the time of the competition, the LKB was not yet extended with collocates (resource 4 of Section 2.1).

Interestingly enough, while the March 2004 competition was running, we could detect, thanks to the interconnection patterns produced by SSI, several inconsistencies in the so-called *golden gloss* data. For example, one of the highest performing sense tagging patterns in the grammar *G* is the *direct hyperonymy* path. This is a hyperonymy path of length 1 between a synset and a sense associated to a word in its gloss. This pattern suggests the correct sense choice with almost 100 percent precision.

An example is $custom\#4$ defined as "habitual patronage." We have that:

$$\{custom\#4\} \xrightarrow{kind\_of} \{trade, patronage\#5\},$$

therefore, we select sense #5 of *patronage*, while the "golden" sense is #1. As a second example, consider $footage\#1$, defined as "the amount of film that has been shot." Here, *film* refers to sense #5, i.e., "photographic material consisting of a base of celluloid," supported by the following path:

$$\{footage\#1\} \xrightarrow{kind\_of} \{film\#5, photographic\_film\},$$

while the "golden" tag is $film\#1$, defined as "a form of entertainment [...]." We do not intend to dispute whether the "questionable" sense assignment is the one provided in the golden gloss or, rather, the hyperonym selected by the WordNet lexicographers. In any case, the detected patterns show a clear inconsistency in the data.

These inconsistent golden glosses (315) have been submitted to the organizers, along with the interconnecting patterns supporting the existence of an inconsistency. Their final decision was to remove these glosses from the data set.

This is per-se an encouraging result: A clear advantage of SSI is the supporting evidence for a sense choice, which makes this algorithm a useful tool to support the task of human annotators.

The SSI version that participated in Senseval-3 has been recently upgraded with an extended LKB, but, as we said, in March 2004, these extensions were not available.

Table 1 provides a summary of the results, including those of our best competitors (first and third) in the actual competition, i.e., TALP research center and Language Computer Corporation. The table includes the results of the "old" SSI with baseline (e.g., with first sense heuristic)

---

22. The first sense heuristic was a common choice for many systems to fulfill the requirement of high recall [30].

TABLE 1
Results of Gloss Disambiguation Task at Senseval-3

System	Prec.	Recall	Attempted
SSI+baseline	0.685	0.684	99.9
SSI standard	0.826	0.323	39.1
TALP Research Center	0.702	0.698	99.9
LanguageComputerCorp	0.721	0.516	71.6

TABLE 2
Precision and Recall by Syntactic Category

	Nouns	Verbs	Adj.
Prec.	86.0%	69.4%	78.6%
Recall	44.7%	13.5%	26.2%

and without baseline since, in Senseval tasks, it is possible to submit up to three runs for each system.

The table shows that SSI obtained the second best result, very close to the first and well over the third. It also shows that, by its very nature, the algorithm is tuned to work with high precision, possibly low recall. The recall critically depends on the semantic closeness of the contexts to be disambiguated, as also clarified by the experiment in Section 5.2.

An additional problem is that the LKB includes rich information for nouns and adjectives, but is very poor for verbs, as shown in Table 2,[23] especially because of an analogous deficiency in WordNet. This, in part, improved in the extended version on LKB since there are several *context* relations for verbs, but needs to be further enhanced in our future work.

Notice that, for the gloss disambiguation task, no training data were available for machine learning WSD methods; consequently, these systems performed poorly in the competition. In fact, the TALP system is also a knowledge-based system.

5.2 Evaluation of the Disambiguation of Collocations Task

This section describes an experiment in which, first, we applied SSI to the task of disambiguating a set of collocations. Then, we enriched the lexical knowledge base with a manually verified set of context associations, extracted from the full set of available collocations.

Finally, we repeated the disambiguation step over the same set of collocations as before to evaluate the trend of improvement that we might obtain with a further extension of the LKB.

We identified 70 sets of collocations of different sizes (one for each selected representative concept), containing a total number of 815 terms to be disambiguated. These terms were manually disambiguated by two annotators, with adjudication in case of disagreement.[24] The application of SSI to the 70 collocations led to a precision result of 85.23 percent and a recall of 76.44 percent. We also analyzed the outcome with respect to different context sizes. The results, reported in Table 3a, show that both recall and precision measures tend to grow with the context size $|T|$. The intuition for this behavior

TABLE 3
Performances of (a) Simple and (b) Enriched SSI* on
Different Context Sizes ($|T|$)

		$|T|=5$	$|T|=10$	$|T|=20$	$|T|=30$	$|T|=40$
Tot # terms:		**175**	**170**	**160**	**150**	**160**
(a)	Recall	66.86%	75.29%	78.75%	80.00%	82.52%
	Prec.	82.98%	82.58%	86.90%	86.96%	86.84%
(b)	Recall	75.43%	82.94%	83.13%	84.00%	88.13%
	Prec.	84.08%	83.95%	90.48%	86.30%	89.81%

is that larger contexts provide richer (and more expressive) semantic interconnections.

Notice that, with respect to other WSD tasks discussed in this section, these disambiguation contexts contain terms with stronger interconnections because collocations express a form of tight semantic relatedness. This explains the high precision results obtained with medium-size or large contexts (about 86.9 percent on average when $20 \leq |T| \leq 40$).

Then, we enriched the SSI lexical knowledge base with about 10,000 manually disambiguated *context* relations. We indicate the enriched version with SSI*.

In order to measure the improvement obtained on the same task as in Table 3, relations connecting concepts in the test set of 70 sets of collocations were excluded (35 percent over a total number of 10,000 relation instances, about 11 relations per representative concept on average). The total number of "survived" relations actually used in the experiment was then 7,000. Such relations concerned 883 representative concepts.

The second experiment resulted in a significant improvement in terms of recall (82.58 percent on average) with respect to the first run, while the increase in precision (86.84 percent, i.e., about +1.6 percent) is not striking. Table 3b shows that both measures tend to increase with respect to the first experiment for all context sizes $|T|$ (with a minor, but still significant increase for larger contexts).

5.3 Evaluation of the Ontology Learning Task

Ontology learning is the task for which SSI was initially conceived and, consequently, the best tailored task. In SSI, the ability to disambiguate a term t critically depends on the semantic closeness, measured by semantic interconnection weights, between t and its context T. In specific domains, like those in which we experimented with on the OntoLearn system, words tend to exhibit much closer semantic relations than in generic sentences (see, for example, Section 5.4).

Furthermore, many interesting applications of WSD, like semantic annotation, intelligent information access, and ontology learning, have a potential impact precisely on specific, technical domains relevant for Web communities willing to share documents, data, and services through the Web.

So far, OntoLearn has been experimented with on several projects, specifically the Harmonize[25] EC project on tourism interoperability, a national project on Web learning[26] in the computer network area, and two still on-going projects, the INTEROP EC NoE[27] on enterprise interoperability and an Italian project (*Legenda*) on ontologies for cultural heritage. Furthermore, we tested also SSI on a financial domain.

---

23. This table is also relative to the version of SSI with which we actually participated in Senseval-3.

24. In this and other evaluation tasks for which no professionally annotated data were available, annotation has been performed by the two authors. In some domains, there were three annotators.

25. Harmonise EC project IST-2000-29329, http://dbs.cordis.lu.

26. http://www.web-learning.org.

27. INTEROP NoE IST-2003-508011, http://interop-noe.org.

We briefly describe here four experiments. In these experiments, SSI is enriched with the 10,000 context relations mentioned in the previous section.

5.3.1   Interoperability

A preliminary task in the first year of the INTEROP project was to obtain a sort of partially structured glossary, rather than an ontology, i.e., a forest of term trees, where, for each term, the following information has to be provided: *definition* of the term, *source* of the definition (domain specialist or Web site), *kind-of relation*, e.g.:

interoperability: *The ability of information systems to operate in conjunction with each other encompassing communication protocols, hardware software, application, and data compatibility layers.*
source: www.ichnet.org/glossary.htm.
kind-of: ability.

The project partners collected through the INTEROP collaborative platform a corpus of relevant documents on enterprise interoperability, under the different perspectives of ontology, architecture, and enterprise modeling. From this set of documents and from online available glossaries in related domains,[28] we extracted about 500 definitions that were then verified by a team of experts, leading to a final glossary of 376 terms.[29]

To arrange terms in term trees, we used the procedure described in Section 4.3, using the SSI algorithm to attach subtrees to WordNet nodes.

First, the definitions in the glossary have been parsed and the word, or multiword expression, representing the hyperonym has been identified. Given the limited number of definitions, we verified this task manually, obtaining a figure of 91.76 percent precision, in line with previous evaluations of the same task.

Overall, the definitions were grouped in 125 subtrees, of which 39 include only two nodes, 43 with three nodes, and the others with more than three nodes. Examples of two term trees are shown in Fig. 6.

In Fig. 6, the collocation of the term *system* might seem inappropriate since this term has a very generic meaning. However, the definition of system in the interoperability glossary is quite specific: "*a set of interacting components for achieving common objectives,*" which justifies its collocation in the tree. A similar consideration applies to service in the second tree.

To evaluate the precision of SSI in assigning each subtree to the appropriate WordNet node, we manually selected the "appropriate" synset for the 125 subtrees roots and then we ran the SSI algorithm as described in Section 4.3.6. To augment the contexts T, we augmented T with the words in the natural language definitions of the glossary terms that belong to a subtree, limited to those below the root node (e.g., for the second subtree of Fig. 6: *software capability, functionality, business capability, competency*).

5.3.2   Computer Networks

In a national project on e-learning, the objective was to annotate the relevant domain terms in available computer
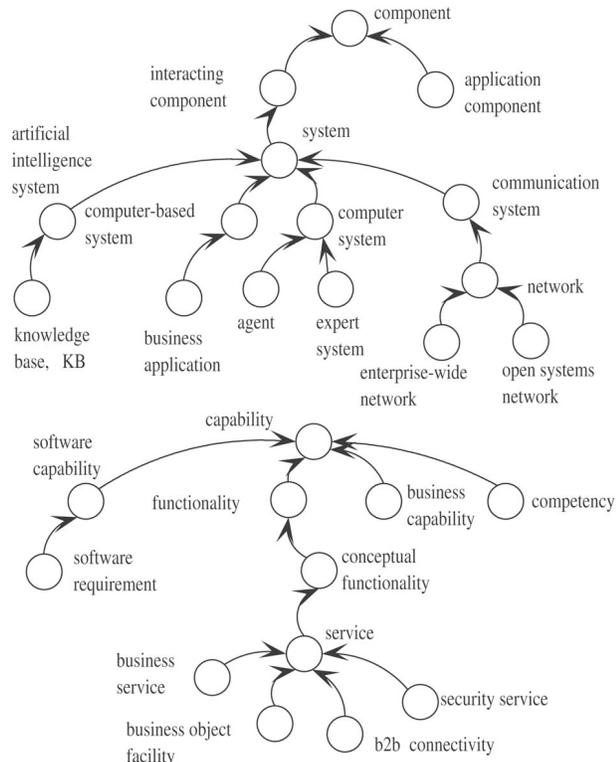

Fig. 6. Subtrees extracted from the Interoperability domain.

network courseware with the concepts of a domain ontology. The ontology was extracted mainly by processing the natural language definitions of a computer network glossary.[30] The performance of hyperonym extraction was 95.5 percent, estimated on a fragment of 200 definitions. The hyperonym information was used to build the subtrees, of which two examples were shown in Fig. 5. The gloss parsing algorithm extracted 655 subtrees, with an average of about 10 terms per tree.

Finally, the subtrees were attached to WordNet nodes. To estimate the precision and recall of node attachments, we manually verified 100 such attachments.

5.3.3   Tourism

The Tourism domain is a collection of hotel descriptions, mainly extracted from the Internet, used during the Harmonize EC project. The number of trees is 44, with an average of 11 terms per tree. We manually labeled 453 terms.

5.3.4   Finance

This domain is the Wall Street Journal corpus, featuring one million words of 1989 Wall Street Journal material. The number of trees is 106, with an average of 18 terms per tree. We manually labeled 638 terms.

In the Tourism and Finance domain, the experiment was sensibly different. Rather than disambiguating only the root, we attempted a disambiguation of the full tree. The reason is that while, in Computer Network and Interoperability, many of the terms in a tree are very technical, therefore the "appropriate" sense in WordNet is simply absent, Tourism

---

28. Interoperability is a new field, therefore many specific definitions were automatically extracted from tutorials and seminal papers made available by the partners. Other definitions were found in related glossaries.
29. Since detailed state-of-the-art and new documents significantly enriched the INTEROP document repository, the glossary acquisition process needs to be repeated in year 2.

30. www.bandwidthmarket.com/resources/glossary/T5.html.

TABLE 4
Performances of SSI* as a Function of $|T|$

Domain	$|T| < 35$		$35 <= |T| < 65$		$|T| >= 65$	
	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.
Finance	78.57%	58.51%	81.82%	63.28%	80.29%	55.46%
Tourism	80.29%	55.46%	80.00%	73.20%	-	-
Networks	80.60%	62.07%	87.5%	87.5%	80.00%	80.00%
Interop.	70.49%	71.66%	-	-	-	-

TABLE 5
Results of the All Words Task in Senseval-3

System	"With U"	"Without U"	
	Prec/Rec	Precision	Recall
GAMBL-AW-S	0.652	0.651	0.651
Sense Learner-S	0.646	0.651	0.642
IRST-DDD-00-U	0.583	0.583	0.582
SSI*	0.604	0.604	0.604

and Finance are midtechnical and have a better correspondence with WordNet. For example, consider the word *block* in *block_transfer_failure* (one of the trees in Fig. 5). This word has 12 senses in WordNet, none of which is appropriate.

Table 4 reports the precision and recall of SSI for the four domains. Performances are shown as a function of the dimension of the context T.

A first remark is that, as already shown in Section 5.2, large contexts increase the chance of finding semantic interconnections. Second, midtechnical domains perform better than highly technical ones. This seems to be a problem with the use of WordNet since many concepts in WordNet do not reflect the specificity of a domain. Better results could be obtained enriching the LKB with a Domain Core Ontology (e.g., the already mentioned CRM-CIDOC), but, unfortunately, few of these ontologies are available. Currently, there are many ongoing efforts to build such Core Ontologies in many domains [31]. An additional problem with the INTEROP domain was the prevailing number of very small contexts (50 percent of the trees have ≤ 3 nodes). As we mentioned, the glossary needs to be extended and this, hopefully, will enrich the domain forest.

5.4 Evaluation of the Senseval-3 All-Words Task

A "classic" task in Senseval competitions is *English All Words*. In Senseval-3, the test data consisted of approximately 5,000 words of running texts from two Wall Street Journal articles and an excerpt of the Brown Corpus. A total of 2,212 words were manually annotated, with a reported interannotator agreement of 72.5 percent. This low value demonstrates the inherent difficulty of the task.

We did not actually participate in this task, but we repeat here the experiment for the purpose of this paper, using the enriched SSI*.

No training data were available for Senseval-3, but systems were allowed to train word sense classifiers on previous Senseval-2 and 1 *All Words* data, as well as on other sense-tagged English texts, including training data provided for other Senseval tasks, e.g., *Lexical Sample*.

We did <u>not</u> train SSI with any additional information concerning the words included in the test set, such as, other articles from the Wall Street Journal. We used the extended LKB, but no ad hoc extensions. Therefore, we consider the system untrained.

One of the problems with the All Words task is that test data include many short sentences with almost unrelated words, like: "*He wondered how he got tied up with this stranger*." This is more frequent in the Brown Corpus sentences, while the Wall Street journal articles include less generic contexts.

In order to increase the dimension of the context T used to start the disambiguation process, we included in each T the words from three contiguous sentences of the test data.[31]

The results are reported in Table 5, along with those of the best supervised and best unsupervised systems participating in the March 2004 competition. As for the gloss disambiguation task (Section 5.1), the partners were requested to maximize both precision and recall, therefore it was necessary to use a baseline.

The organizers provided two types of evaluation. In the "with U" evaluation, they assumed an answer U (*untaggable*) whenever the system failed to provide a sense for a word. Since certain words were marked as untaggable also in the test data, an instance would be scored as correct if it was marked with a U and the test data also marked it with a U.

The "without U" evaluation simply skips every word tagged with a U. Thus, untagged words do not affect the precision, but lower the recall.

Table 5 shows, for comparison, the results obtained by the first two supervised systems (marked with S) and the first untrained (marked with U). As previously argued, SSI is to be considered as untrained.

The table shows that SSI* performs better than the best untrained system. Again, the performance for nouns and adjectives was considerably higher than for verbs.

6 FINAL REMARKS

Word Sense Disambiguation is perhaps the most complex natural language processing task. A structural approach to sense classification such as the one presented in this paper seems particularly well-suited, for a variety of reasons:

- Structured features to represent word senses can be extracted from available online lexical and semantic resources. Though an integration effort is certainly needed, we can foresee that, since more and more resources are being made available, better performances will be achieved by a structural approach that relies on these features.
- Structured relations (i.e., interconnection patterns) among word senses, detected by the SSI algorithm, provide a readable, and very interesting, basis for a variety of automatic tasks, such as ontology learning, query expansion, parsing of dictionary definitions, etc., as well as being a guide for human annotators. In fact, certain detected interconnection patterns provide very strong clues for manual sense tagging, as suggested during the Senseval-3 gloss parsing experiment [28] and in [32].

31. This implies that, if a word occurs more than one time, we choose the same sense.

- The algorithm performs significantly better when disambiguating large contexts, in midtechnical and sufficiently focused domains. In unfocused contexts (e.g., the Brown corpus sentences in Senseval-3), words do not exhibit strong semantic relations. In overly technical domains, the algorithm suffers from some inadequacy of the "base" ontology (e.g., Word-Net) that should be replaced by a Core Domain Ontology.

SSI is an open research area in our group and several improvements are being explored. The algorithm can be improved both through further enrichments of the LKB, as new resources become available, and through a refinement and extension of the grammar G.

In the current LKB, limited information is encoded for verbs as a consequence of a limitation in WordNet. Better resources are available for verbs, but, again, an integration effort is necessary.

In the current version of G, grammar rules seek 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, the path weighting method (currently a perceptron) can be replaced by a more sophisticated technique.

ACKNOWLEDGMENTS

REFERENCES

[1] N. Ide and J. Veronis, "Introduction to the Special Issue on Word Sense Disambiguation," *Computational Linguistics,* vol. 24, no. 1, Mar. 1998.

[2] Y. Wilks, "A Preferential Pattern-Seeking Semantics for Natural Language Inference," *Artificial Intelligence,* vol. 6, pp. 53-74, 1978.

[3] R. Schank and R. Abelson, *Scripts, Plans, Goals, and Understanding.* Hillsdale, N.J.: Lawrence Erlbaum, 1977.

[4] R. Mihalcea and D.I. Moldovan, "A Highly Accurate Bootstrapping Algorithm for Word Sense Disambiguation," *Int'l J. Artificial Intelligence Tools,* vol. 10, nos. 1-2, pp. 5-21, 2001.

[5] P. Resnik, "Using Information Content to Evaluate Semantic Similarity In A Taxonomy," *Proc. Int'l Joint Confs. Artificial Intelligence (IJCAI),* 1995.

[6] R. Krovetz and W.B. Croft, "Word Sense Disambiguation Using Machine Readable Dictionaries," *Proc. 12th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 127-136, 1989.

[7] D. Yarowski, "Word-Sense Disambiguation Using Statistical Models of the Roget's Categories Trained on Large Corpora," *Proc. 14th Int'l Conf. Computational Linguistics (COLING-92),* pp. 454-460, 1992.

[8] W. Gale, K. Church, and D. Yarowsky, "One Sense per Discourse," *Proc. DARPA Speech and Natural Language Workshop,* pp. 233-237, Feb. 1992.

[9] W. Gale, K. Church, and D. Yarowsky, "A Method for Disambiguating Word Senses in a Corpus," *Computer and the Humanities,* vol. 26, pp. 415-439, 1992.

[10] J. Gonzalo, F. Verdejo, I. Chugur, and J. Cigarr'an, "Indexing with WordNet Synsets Can Improve Text Retrieval," *Proc. COLING/ ACL '98 Workshop Usage of WordNet for NLP,* 1998.

[11] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific Am.,* May 2001.

[12] OpenCyc home page, http://www.opencyc.org/, 2005.

[13] *WordNet: An Electronic Lexical Database,* C. Fellbaum, ed. MIT Press, 1998.

[14] FrameNet home page, http://www.icsi.berkeley.edu/framenet/, 2005.

[15] K.S. Fu, *Syntactic Pattern Recognition and Applications.* Englewood Cliffs, N.J.: Prentice Hall, 1982.

[16] *Syntactic and Structural Pattern Recognition: Theory and Applications,* H. Bunke and A. Sanfeliu, eds. World Scientific, 1990.

[17] N. Guarino and C. Welty, "Evaluating Ontological Decisions with OntoClean," *Comm. ACM,* vol. 45, no. 2, pp. 61-65, 2002.

[18] Second Global WordNet Conf., Jan. 2004, http://www.fi.muni. cz/gwc2004/.

[19] B. Magnini and G. Cavaglià, "Integrating Subject Field Codes into WordNet," *Proc. Second Int'l Conf. Language Resources and Evaluation (LREC2000),* 2000.

[20] G.A. Miller, M. Chodorow, S. Landes, C. Leacock, and R.G. Thomas, "Using a Semantic Concordance for Sense Identification," *Proc. ARPA Human Language Technology Workshop,* pp. 240-243, 1994.

[21] H.T. Ng and H.B. Lee, "Integrating Multiple Knowledge Sources to Disambiguate Word Sense: An Exemplar-Based Approach," *Proc. 34th Ann. Meeting Assoc. for Computational Linguistics,* 1996.

[22] *Oxford Collocations,* D. Lea, ed. Oxford Univ. Press, 2002.

[23] *Longman Language Activator,* K. Longman, ed. Pearson Education, 2003.

[24] R. Navigli, "Semi-Automatic Extension of Large-Scale Linguistic Knowledge Bases," *Proc. 18th FLAIRS Int'l Conf.,* May 2005.

[25] R. Mihalcea and D. Moldovan, "eXtended WordNet: Progress Report," *Proc. Workshop WordNet and Other Lexical Resources,* 2001.

[26] I.H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools with Java Implementations.* San Francisco: Morgan Kaufmann, 2000.

[27] R. Navigli and P. Velardi, "Learning Domain Ontologies from Document Warehouses and Dedicated Web Sites," *Computational Linguistics,* MIT Press, June 2004.

[28] K. Litkowski, "Senseval-3 Task: Word Sense Disambiguation of WordNet Glosses," *Proc. Senseval-3 Third Int'l Workshop Evaluation of Systems for Semantic Analysis of Texts,* pp. 13-16, July 2004.

[29] R. Navigli and P. Velardi, "An Analysis of Ontology-Based Query Expansion Strategies," *Proc. Workshop Adaptive Text Extraction and Mining,* Sept. 2003.

[30] D. McCarthy, R. Koeling, J. Weeds, and J. Carroll, "Finding Predominant Word Senses in Untagged Text," *Proc. 42nd Ann. Meeting Assoc. for Computational Linguistics,* July 2004.

[31] *Proc. EKAW*04 Workshop Core Ontologies in Ontology Eng.,* http:// sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-118/, 2004.

[32] R. Navigli, "Supporting Large-Scale Knowledge Acquisition with Structural Semantic Interconnections," *Proc. AAAI Spring Symp.,* 2005.

Roberto Navigli received the Laurea degree in computer science from the University of Roma "La Sapienza." He is a PhD student in the Department of Computer Science of the University of Roma. His research interests include natural language processing and knowledge representation.

Paola Velardi received the Laurea degree in electrical engineering from the University of Roma "La Sapienza". She is a full professor in the Department of Computer Science at the University of Roma "La Sapienza." Her research interests include natural language processing, machine learning, and the semantic Web.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.