Structural Semantic Interconnection: a knowledge-based approach to Word Sense Disambiguation

Roberto NAVIGLI
Dipartimento di Informatica,
Università di Roma "La Sapienza"
Via Salaria, 113 - 00198 Roma, Italy
navigli@di.uniroma1.it

Paola VELARDI
Dipartimento di Informatica,
Università di Roma "La Sapienza"
Via Salaria, 113 - 00198 Roma, Italy
velardi@di.uniroma1.it

Abstract

In this paper we describe the SSI algorithm, a structural pattern matching algorithm for WSD. The algorithm has been applied to the gloss disambiguation task of Senseval-3.

1 Introduction

Our approach to WSD lies in the *structural pattern recognition* framework. Structural or syntactic pattern recognition (Bunke and Sanfeliu, 1990) 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 Building structural representations of word senses

We build a structural representation of word senses using a variety of knowledge sources, i.e. WordNet, Domain Labels (Magnini and Cavaglia, 2000), annotated corpora like SemCor and *LDC-DSO*[1]. We use this information to automatically generate labeled directed graphs (*digraphs*) 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 oval in Figure 1. This distance has been experimentally established.

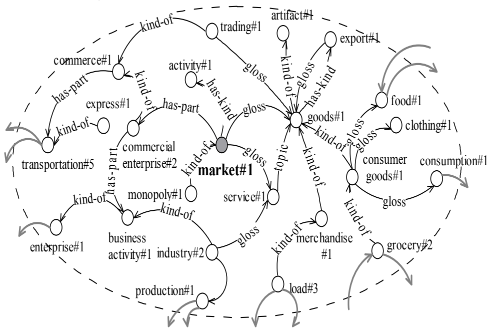

Figure 1. Graph representations for sense #1 of *market.*

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

3 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 $I = [S^{t_1}, \ldots, S^{t_n}]$, i.e. the semantic interpretation of T, where S^{t_i} [2] is either the chosen sense for t_i (i.e., the result of a previous

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

[2] Note that with S^{t_i} we refer interchangeably to the semantic graph associated with a sense or to the sense *name*.

disambiguation step) or the empty set (i.e., the term is not yet disambiguated).

A set of *pending* terms is also maintained, $P = \{t_i \mid S^{t_i} = \varnothing\}$. 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 the specific WSD task considered. Section 5 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 $f_I : CxT \rightarrow \Re$, where C is the set of all the concepts in the ontology O, defined as follows:

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

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 ($\rho'$) 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 $A \rightarrow \alpha$ 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 $\arg\max_{S \in C} f_I(S,t)$ as the most likely interpretation of t and updates the list I with the chosen concept. A threshold can be applied to $f(S,t)$ 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 $f_I(S,t) > 0$, 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.

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 initialization policy selects a term $t \in T$ and the execution is forked into as many processes as the number of senses of t.

4 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 e_i and e_{i+1} are incoming and/or outgoing from the same concept node, that is $\xrightarrow{e_i}(S)\xrightarrow{e_{i+1}}$, $\xleftarrow{e_i}(S)\xrightarrow{e_{i+1}}$, $\xrightarrow{e_i}(S)\xleftarrow{e_{i+1}}$, $\xleftarrow{e_i}(S)\xleftarrow{e_{i+1}}$. 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.

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, trained with standard word sense disambiguation data, such as the SemCor corpus.

Examples of the rules in G are provided in the subsequent Section 5.

5 Application of the SSI algorithm to the disambiguation of WordNet glosses

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* }[3]

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[4]:

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*

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) 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 Evaluation

The SSI algorithm is currently tailored for noun disambiguation. Additional semantic knowledge and ad-hoc rules would be needed to detect semantic patterns centered on concepts associated to verbs. Current research is directed towards integrating in semantic graphs information from FrameNet and VerbNet, but the main problem is harmonizing these knowledge bases with WordNet's senses and relations inventory. A second problem of SSI, when applied to unrestricted WSD tasks, is that it is designed to disambiguate with high precision, possibly low recall. In many interesting applications of WSD, especially in information retrieval, improved document access may be obtained even when only few words in a query are disambiguated, but the disambiguation precision needs to be well over the 70% threshold. Supporting experiments are described in (Navigli and Velardi, 2003).

The results obtained by our system in Senseval-3 reflect these limitations (see Figure 2).

The main run, named OntoLearn, uses a threshold to select only those senses with a weight

[3] For convenience here we denote I as a set rather than a list.

[4] With $S \xrightarrow{R}^{i} S'$ we denote a path of i consecutive edges labeled with the relation R interconnecting S with S'.

over a given threshold. OntoLearnEx uses a non-greedy version of the SSI algorithm. Again, a threshold is used to accepts or reject sense choices. Finally, OntoLearnB uses the "first sense" heuristics to select a sense, every since a sense choice is below the threshold (or no patterns are found for a given word).

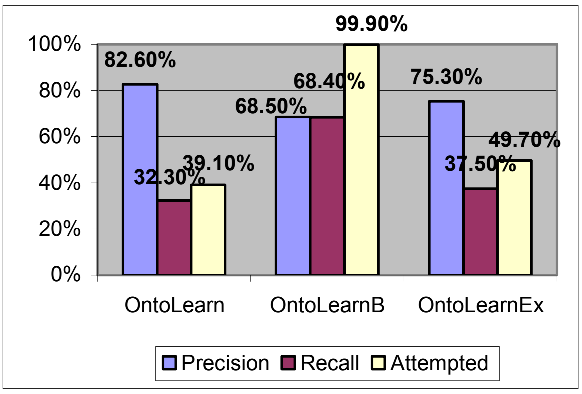

Figure 2. Results of three runs submitted to Senseval-3.

Table 1 shows the precision and recall of OntoLearn main run by syntactic category. It shows that, as expected, the SSI algorithm is currently tuned for noun disambiguation.

	Nouns	Verbs	Adj.
Precision	86.0%	69.4%	78.6%
Recall	44.7%	13.5%	26.2%
Attempted	52.0%	19.5%	33.3%

Table 1. Precision and Recall by syntactic category.

The official Senseval-3 evaluation has been performed against a set of so called "golden glosses" produced by Dan Moldovan and its group[5]. This test set however had several problems, that we partly detected and submitted to the organisers.

Besides some technical errors in the data set (presence of WordNet 1.7 and 2.0 senses, missing glosses, etc.) there are sense-tagging inconsistencies that are very evident.

For example, one of our highest performing sense tagging rules in SSI is the *direct hyperonymy* path. This rule reads as follows: "if the word w_j appears in the gloss of a synset S_i, and if one of the synsets of w_j, S_j, is the direct hyperonym of S_i, then, select S_j as the correct sense for w_j".

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

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

therefore we select sense #5 of patronage, while Moldovan's "golden" sense is #1.

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 patterns (313) have been submitted to the organisers, who then decided to remove them from the data set.

7 Conclusion

The interesting feature of the SSI algorithm, unlike many co-occurrence based and statistical approaches to WSD, is a *justification* (i.e. a set of semantic patterns) to support a sense choice. Furthermore, each sense choice has a weight representing the confidence of the system in its output. Therefore SSI can be tuned for high precision (possibly low recall), an asset that we consider more realistic for practical WSD applications.

Currently, the system is tuned for noun disambiguation, since we build structural representations of word senses using lexical knowledge bases that are considerably richer for nouns. Extending semantic graphs associated to verbs and adding appropriate interconnection rules implies harmonizing WordNet and available lexical resources for verbs, e.g. FrameNet and VerbNet. This extension is in progress.

References

H. Bunke and A. Sanfeliu (editors) (1990) *Syntactic and Structural pattern Recognition: Theory and Applications* World Scientific, Series in Computer Science vol. 7, 1990.

A. Gangemi, R. Navigli and P. Velardi (2003) "The OntoWordNet Project: extension and axiomatization of conceptual relations in WordNet", 2nd *Int. Conf.* ODBASE, ed. Springer Verlag, 3-7 November 2003, Catania, Italy.

B. Magnini and G. Cavaglia (2000) "Integrating Subject Field Codes into WordNet", *Proceedings of* LREC2000, Atenas 2000.

Milhalcea R., Moldovan D. I. (2001) "eXtended WordNet: progress report". *NAACL 2001 Workshop on WordNet and other lexical resources*, Pittsburg, June 2001.

Navigli R. and Velardi P. (2003) "An Analysis of Ontology-based Query Expansion Strategies", *Workshop on Adaptive Text Extraction and Mining* September 22nd, 2003 Cavtat-Dubrovnik (Croatia), held in conjunction with ECML 2003.

[5] http://xwn.hlt.utdallas.edu/wsd.html