An Endogeneous Corpus-Based Method for Structural Noun Phrase Disambiguation

Didier Bourigault
Centre d'Analyse et de Mathématiques Sociales
(EHESS - Paris Sorbonne - CNRS)
and
Electricité de France - Direction des Etudes et Recherches
Service Informatique et Mathématiques Appliquées
1, avenue du Général de Gaulle
92141 Clamart Cedex
FRANCE

Abstract

In this paper, we describe a method for structural noun phrase disambiguation which mainly relies on the examination of the text corpus under analysis and doesn't need to integrate any domain-dependent lexico- or syntactico-semantic information. This method is implemented in the Terminology Extraction Sotware LEXTER. We first explain why the integration of LEXTER in the LEXTER-K project, which aims at building a tool for knowledge extraction from large technical text corpora, requires improving the quality of the terminolgy extracted by LEXTER. Then we briefly describe the way LEXTER works and show what kind of disambiguation it has to perform when parsing "maximal-length" noun phrases. We introduce a method of disambiguation which relies on a very simple idea : whenever LEXTER has to choose among several competing noun sub-groups in order to disambiguate a maximal-length noun phrase, it checks each of these sub-groups if it occurs anywhere else in the corpus in a non-ambiguous situation, and then it makes a choice. The half-a-million words corpus analysis resulted in an efficient strategy of disambiguation. The average rates are :

27 %	no disambiguation
70 %	correct disambiguation
3 %	wrong disambiguation

1 The LEXTER-K project : knowledge extraction from large technical text corpora

LEXTER is a Terminology Extraction Software (Bourigault, 1992a, 1992b). A corpus of French-language texts on any (technical) subject is fed in. LEXTER performs a grammatical analysis of this corpus and yields a list of noun phrases which are likely to be terminological units, representing the concepts of the subject field. This list together with the corpus it has been extracted from is then passed on to an expert for validation by the means of a terminological hypertext web. LEXTER has been developped in an industrial context, in the Research and Development Division of Electricité de France. It was previously designed to deal with the problem of creating or updating thesauri used by an Automatic Indexing System.

We are integrating LEXTER in a text analysis tool to aid knowledge acquisition in the framework of Knowledge-Based System construction. This tool (LEXTER-K) will propose a structured list of candidate terms, rather than a flat list, which could be considered as a first coarse-grained modelisation of the information conveyed by the texts under analysis.

Structuring of the terminology will be performed in two ways : on the one hand, by a structural analysis of the terminological noun phrases extracted by LEXTER; on the other hand, by an analysis of the sentences in which the candidate terms occur. This analysis will focus on the most relevant terms, determined by a statistical processing based on the assumption that the most frequent terms are probably the most relevant.

We plan a two-stage architecture for LEXTER-K, that is, (1) the extraction of the terminology of the subject field, by a robust grammatical analysis (LEXTER), (2) the syntactic analysis of the sentences by a parser using this terminology. The syntactic structures of the sentences in a text, and the syntactic structures of the terminological units have to be placed on two different organisational levels. As the terminological unit is a semantic unit, it should be treated as such on the syntactic level, as well. Dissociating these two

analysis, though one taking advantages of the results given by the other, will guarantee a better efficiency for the parser, in particular by limiting the combinatory explosion of structural ambiguities.

It is well known that Natural Language systems usually require considerable knowledge acquisition, especially in building a specifically oriented field vocabulary in the case of systems which have to analyse technical texts. We think that this two-stage analysis (extracting the terminology of the domain with a robust superficial analysis, and analysing the texts with a more in-depth parser using this terminology), may lighten the expensive burden of hand-coding a specialized language and may lead to more generic and domain-independant Natural Language systems.

As long as the terminology extracted during the first step has not been validated by an expert, as is now the case, and it directly feeds the syntactic analyser, this two-stage architecture requires a better quality of the terminology extracted by LEXTER. This is the reason why we tried to improve the precision rate in the detection of the terminological noun phrases by implementing an efficient strategy for structural noun phrase disambiguation. This strategy is described in the following sections.

2 The issue of parsing the ambiguous "Maximal-Length" Noun Phrases

In this section, we briefly describe the type of grammatical analysis performed by LEXTER to extract likely terminological units and show what kind of disambiguation LEXTER has to perform.

As we already pointed out, LEXTER has been achieved in an industrial context; from the beginning of the project, we had decided to focus upon a strongly restrictive criterium : applying and testing the system over a wide range of texts. The texts to be analysed are unrestricted texts gathered in large corpora. We had then to choose a fast and well-proved method. Moreover, we argued that, given the restricted grammatical structures of complex terminological units, it was not necessary to go into a complete syntactic analysis of the sentences to extract the terminology from a corpus (Bourigault, 1992b).

First, a morphological analyser tags the texts, using a large lexical database and rules of lexical disambiguation. LEXTER treats texts in which each word is tagged with a grammatical category (noun, verb, adjective, etc.). LEXTER works in two main phases : (1) splitting and (2) parsing.

(1) At the splitting stage, LEXTER takes advantage of "negative" knowledge about the form of terminological units, by identifying those string level patterns which never go to make up these units and which can thus be considered as potential terminological limits. Such patterns are made up by, say, conjugated verbs, pronouns, conjonctions, certain strings of preposition + determiner. The splitting module is thus set up with a base of about 60 rules for identifying frontier markers, which it uses to split the texts. The splitting phase produces a series of text sequences, most often noun phrases. These noun phrases may well be likely terminological units themselves, but more often than not, they contain sub-groups which are also likely units. That is why it is preferable at the splitting stage to refer to the noun phrases identified as "maximal-length noun phrases". Here is an example of a real maximal-length noun phrase : *MESURE DU DEBIT DU VENTILATEUR D'EXTRACTION AVEC TRAPPE EN POSITION FERMEE* (*noun prep det noun prep det noun prep noun prep noun prep noun adj*).

(2) At the parsing stage, LEXTER parses the maximal-length noun phrases (henceforth MLNP) in order to generate sub-groups, in addition to the MLNP, which are likely terminological units by virtue of their grammatical structure and their position in the MLNP. The LEXTER parsing module is made up of parsing rules which indicate which sub-groups to extract from a MLNP on the basis of grammatical structure.

Some of the MLNP structures are non-ambiguous : given such a structure it can be stated with a very high rate of certainty (Bourigault 1992a) that only one parsing is valid. The corresponding parsing rules are called non-ambiguous rules. For example, structure (1) is non-ambiguous, and parsing rule [a] is a non-ambiguous rule.

(1) *noun1 adj prep noun2*

parsing rule [a]
noun1 adj prep noun2
-->
noun1 adj

example of parse
FUSIBLE THERMIQUE DE FERMETURE
-->
FUSIBLE THERMIQUE

Some of the MLNP structures are ambiguous, that is, given such a structure it cannot be stated with a sufficient rate of certainty that only one parsing is valid. Several sub-structures compete. The corresponding ambiguous parsing rules generate several competing sub-groups. For example, when information about gender or number agreement are not available or of no help, structure (2) is ambiguous, that is, either

the adjective attaches the head *noun1* of the noun sub-group *noun1 prep noun2*, or it attaches *noun2*, constituting the noun sub-group *(noun2 adj)*; the competing noun sub-groups *(noun1 prep noun2)* and *(noun2 adj)* will be generated by the ambiguous parsing rule [b]. Structure (3) and parsing rule [c] are other examples of ambiguous structure and rule.

(2) *noun1 prep noun2 adj*

parsing rule [b]
noun1 prep noun2 adj
-->
 # noun1 prep noun2
 # noun2 adj

example of parse
REGISTRE D'EQUILIBRAGE MANUEL
-->
 # REGISTRE D'EQUILIBRAGE
 # EQUILIBRAGE MANUEL

(3) *noun1 prep noun2 prep noun3*

parsing rule [c]
noun1 prep noun2 prep noun3
-->
 # noun1 prep noun2
 # noun2 prep noun3

example of parse
ALARME DE SYNTHESE DE DEFAUT
-->
 # ALARME DE SYNTHESE
 # SYNTHESE DE DEFAUT

The issue is how to disambiguate in cases of MLNP with ambiguous structures, that means, whenever an ambiguous rule applies, how to choose among the competing generated sub-groups. The strategy of disambiguation is described in the next section.

3 Strategy of disambiguation : looking at non ambiguous situations anywhere else in the corpus

The strategy of disambiguation relies on a very simple idea, that of looking for non-ambiguous situations elsewhere in the corpus. Whenever an ambiguous rule applying to a MLNP with an ambiguous structure generates competing sub-groups, LEXTER (1) checks each of them to ascertain if it has been detected in a non-ambiguous situation (i.e. generated by a non-ambiguous rule) somewhere else in the corpus, and (2) chooses among the competing sub-groups using a set of disambiguation rules.

There is one specific set of disambiguation rules for each ambiguous structure, which covers all the possibles situtations, that is, all, some, only one, none of the competing sub-group non-ambiguously detected.

3.1 Situations where none of the competing sub-groups has been non-ambiguously detected

Given an ambiguous Maximal-Length Noun Phrase, if none of the competing sub-groups has been detected in a non-ambiguous situation, LEXTER proposes only this MLNP, without any sub-group. No disambiguation is performed. On our half-a-million words test corpus, the total number of non-ambiguous MLNPs is 13,591, the total number of ambiguous MLNPs is 3,230, among which 880 are not disambiguated. The average rate of no-disambiguation is 27%. Rates of no-disambiguation for the ten most frequent ambiguous structures are shown in Table 1.

(1)	(2)	(3)	(4)
noun prep noun adj	573	110	19 %
noun prep noun prep det noun	331	91	27 %
noun adj noun	294	74	25 %
noun prep noun noun	260	53	20 %
noun prep noun prep noun	241	55	23 %
noun noun adj	193	73	38 %
noun noun noun	160	47	29 %
noun noun prep noun	82	27	33 %
noun prep noun prep noun adj	42	7	17 %
noun prep noun adj adj	33	2	6 %

Table 1 Rates of no-disambiguation for the ten most frequent ambiguous structures
(1) ambiguous Maximal-Length Noun Phrase (MLNP) structure
(2) total number of MLNP with this structure on the half-a-million words test corpus
(3) number of cases where none of the competing subgroups has been detected in a non-ambiguous situation
(4) rate of no-disambiguation

We are investigating rules that could perform a correct disambiguation in some of these cases. Choosing the right sub-group can be done by checking for each competing sub-group if it has been generated from the analysis of *other* ambiguous MLNP. For example, *REJET D'AIR FROID* and *CIRCUIT D'AIR FROID* are two ambiguous MLNP extracted from the test corpus and parsed by parsing rule [b] above :

REJET D'AIR FROID
-->
 # REJET D'AIR
 # AIR FROID

CIRCUIT D'AIR FROID
-->
 # CIRCUIT D'AIR
 # AIR FROID

Since none of the sub-groups *REJET D'AIR* and *AIR FROID* on the one hand, and *CIRCUIT D'AIR* and *AIR FROID* on the other hand, have been detected in non-ambiguous situations, the MLNPs *REJET D'AIR FROID* and *CIRCUIT D'AIR FROID* have not been disambiguated by LEXTER. But comparing the parsings of these ambiguous MLNPs (*AIR FROID* generated in both cases) can lead to the hypothesis that extracting *AIR FROID* is the correct way of disambiguating them. This hypothesis is reinforced by the fact that the pattern *AIR + adj.* is very productive in the corpus (AIR EXTERIEUR, AIR FRAIS, AIR NEUF, AIR AMBIANT, AIR RECYCLE, etc.).

Our experiments show that such situations (sub-groups never non-ambiguously detected but generated from different ambiguous MLNPs) are very rare and this explains why we have no specific treatment for them yet.

	noun1 prep noun2 detected	***noun1 prep noun2*** not detected
noun2 adj detected	number of occurrences : 141 ***noun2 adj*** number of wrong disamb. : 16	number of occurrences : 132 ***noun2 adj*** number of wrong disamb. : 1
noun2 adj not detected	number of occurrences : 190 ***noun1 prep noun2*** number of wrong disamb. : 15	number of occurrences : 110

Table 2 Set of disambiguation rules for the ambiguous parsing rule [b] :
noun1 prep noun2 adj
->
 # noun1 prep noun2
 # noun2 adj

	noun1 prep noun2 detected	***noun1 prep noun2*** not detected
noun2 prep noun3 detected	number of occurrences : 52 ***noun2 prep noun3*** number of wrong disamb. : 2	number of occurrences : 81 ***noun2 prep noun3*** number of wrong disamb. : 0
noun2 prep noun3 not detected	number of occurrences : 53	number of occurrences : 55

Table 3 Set of disambiguation rules for the ambiguous parsing rule [c] :
noun1 prep noun2 prep noun3
->
 # noun1 prep noun2
 # noun2 prep noun3

3.2 Situations where at least one competing sub-group has been non-ambiguously detected

Given an ambiguous MLNP, systematically keeping (all) the competing sub-group(s) detected elsewhere in the corpus in a non-ambiguous situation is not a satisfying principle of disambiguation. We need more precise rules of disambiguation.

For example, for each of the parsing rules [b] and [c], in more than 20 % of the cases on our test corpus (see top left cells of Table 2 and Table 3), *both* competing sub-groups have been non-ambiguously detected. That means that both sub-groups are attested valid noun phrases as such. However, only one of them corresponds to a correct parsing of the MLNP they have been extracted from. In these cases keeping both sub-groups would alter the precision rate since one of them is not grammatically valid. On the contrary, generating none of them would alter the recall rate. We chose to build a set of disambiguation rules for each of the ambiguous parsing rules.

To work out the disambiguation rules, we adopted an empirical approach based on large-scale corpus experimentation. For each of the ambiguous structures, we examined all the different situations of disambiguation (only one, more than one, all the competing sub-groups non-ambiguously detected) and for each of them, we parsed by hand a significative number of ambiguous noun phrases extracted from a reference test corpus. Applied to the ambiguous parsing rules [b] and [c], this approach led us to the following set of disambiguation rules (see Table 2 and Table 3) :

- where *both* competing sub-groups have been non-ambiguously detected, we checked that most often (125 cases/141 for rule [b], 50 cases/52 for rule [c]) the correct parsing isolates the second sub-group,*noun2 adj* for rule [b], *noun2 prep noun3* for rule [c] (see the top left cells of Table 2 and Table 3).

- where *only the second* sub-group has been non-ambiguously detected, it always corresponds to the correct parsing and so it is systematically kept (see the top right cells of Table 2 and Table 3).

- parsing rules [b] and [c] differs for the situations where *only the first* sub-group (*noun1 prep noun2* for both rules) has been non-ambiguously detected. For rule [b], this sub-group is kept since it most often corresponds to a correct parsing of the MLNP (175 cases/190, see the bottom left cell of Table 2). On the contrary, for rule [c], no systematic rule can be stated since the correct parsing sometimes isolates this non-ambiguously detected sub-group, but often isolates the second one (*noun2 prep noun3*), altough it appears nowhere else in the corpus in a non-ambiguous situation (see the bottom left cell of Table 3). This mainly happens in cases of "elliptical denominations", that is, a concept is first designated in a text by a "complete" term (for example, *CIRCUIT D'ASPERSION D'ENCEINTE*), and then is systematically refered to with an "elliptical" term (for example, *CIRCUIT D'ASPERSION*).

The results we obtained with such sets of desambiguation rules (see Table 4) are satisfactory and show that the strategy described in this paper is efficient. This is partly due to the fact that terminological noun phrases are fixed never disconnected sequences of words with constrained grammatical structures. Our strategy was not designed to deal with adjective and prepositional phrase attachment in unrestricted noun phrases.

rule [b] *noun1 prep noun2 adj* -->> *# noun1 prep noun2* *# noun2 adj*	
rate of no-disambiguation	19 %
rate of correct disambiguation	75 %
rate of wrong-disambiguation	6 %
rule [c] *noun1 prep noun2 prep noun3* -->> *# noun1 prep noun2* *# noun2 prep noun3*	
rate of no-disambiguation	45 %
rate of correct disambiguation	54 %
rate of wrong-disambiguation	1 %

Table 4 Rates of disambiguation for parsing rules [b] and [c]

4 Related works

Some of the methodological ideas of LEXTER are similar to those expressed in (Andreewsky *et al.*, 1977) for the extraction and disambiguation of lexical sequences in a method of building lexical semantic relations dictionaries. More recently a great deal of work in computational linguistics has been devoted to ambiguity resolution. Many rule-based approaches have been proposed which require a huge amount of hand-coded knowledge. (Jensen and Binot, 1987) proposes a method to disambiguate prepositional phrase attachment in English sentences which is similar to ours in that it aims at eliminating the hand coding of semantic information by exploiting an already available source of information. This method differs from ours in that the source of information is a machine-readable dictionary, and it uses complex heuristics, which are different according to the preposition under analysis and

it makes use of some semantic relationships signaled by prepositions.

With regards to the specific problem of structural noun phrase disambiguation, (Wermter, 1989) describes some experiments on article titles for scientific and technical domains. These noun phrases have very similar grammatical structures to those of our "maximal-length noun phrases". The proposed approach, based upon the integration of semantic and syntactic constraints, is quite different from ours since it requires hand-coding of head nouns with semantic features.

(Zernik, 1992a, 1992b) describes a method to distinguish thematic relations (e.g., *expressed concerns*), which take on syntactic variations in the corpus, and sentential relations (e.g., *preferred stocks*) that do not. It consists of testing how many times a given ambiguous relation occurs anywhere else in the corpus in different syntactic configurations. This method ("*Asking the right questions of the corpus*") is similar to the strategy described in this paper, and we agree with Zernik's general line of thinking : "*In order for a program to interpret natural language text, it must train on and exploit word connections in the text under interpretation* (Zernik, 1992a)."

5 Conclusion

We described a method for the structural disambiguation of complex terminological noun phrases, which relies on a simple idea : looking for non-ambiguous situations anywhere else in the corpus. We call this method *Endogeneous* since it does not need to integrate any domain-dependent, lexico- or syntactico-semantic information. We called it *Corpus-Based* since it exploits the experience the system has acquired after a first reading of the corpus under analysis. In that sense, our method can be viewed as a particular implementation of a model of Language Performance (Bod, 1992), and may belong to the family of Data-Oriented methods in Computational Linguistics.

References

[Andreewsky *et al.*, 1977] Alexandre Andreewsky, Christian Fluhr and Fathi Debilli. Computational Learning of Semantic Lexical Relations for the Generation and Automatical Analysis of Content. *Information Processing 77*, 1977

[Bod, 1992] Rens Bod. A Computational Model of Language Performance : Data Oriented Parsing. In *Proceedings of COLING-92*, Nantes, August 1992

[Bourigault, 1992a] Didier Bourigault. LEXTER, un Logiciel d'EXtraction de TERminologie. In *Proceedings of the 2nd symposium of TermNet*, Avignon, May 1992

[Bourigault, 1992b] Didier Bourigault. Surface Grammatical Analysis for the Extraction of Terminological Noun Phrases. In *Proceedings of COLING-92*, Nantes, August 1992

[Jensen and Binot, 1987] Karen Jensen and Jean-Louis Binot. Disambiguating Prepositional Phrase Attachments by Using On-Line Dictionary Definition. *Computational Linguistics 13 (3-4)*, 1987

[Wermter, 1989] Stefan Wermter. Integration of Semantic and Syntactic Constraints for Structural Noun Phrases Disambiguation. In *Proceedings of the 11th IJCAI*, 1989, Detroit

[Zernik, 1992a] Uri Zernik. Shipping Departments vs. Shipping Pacemakers : Using Thematic Analysis to Improve Tagging Accuracy. In *Proceedings of AAAI-92*, July 1992, San Jose

[Zernik, 1992b] Uri Zernik. Closed Yesterday and Closed Minds: Asking the Right Questions of the Corpus To Distinguish Thematic from Sentential Relations. In *Proceedings of COLING-92*, August 1992, Nantes