Clustering Syntactic Positions with Similar Semantic Requirements

Pablo Gamallo*
Universidade de Santiago de Compostela

Alexandre Agustini†
Pontifícia Universidade Católica do Rio Grande do Sul, Centro de Informática e Tecnologias de Informação

Gabriel P. Lopes‡
Centro de Informática e Tecnologias de Informação

This article describes an unsupervised strategy to acquire syntactico-semantic requirements of nouns, verbs, and adjectives from partially parsed text corpora. The linguistic notion of requirement underlying this strategy is based on two specific assumptions. First, it is assumed that two words in a dependency are mutually required. This phenomenon is called here ***corequirement****. Second, it is also claimed that the set of words occurring in similar positions defines extensionally the requirements associated with these positions. The main aim of the learning strategy presented in this article is to identify clusters of similar positions by identifying the words that define their requirements extensionally. This strategy allows us to learn the syntactic and semantic requirements of words in different positions. This information is used to solve attachment ambiguities. Results of this particular task are evaluated at the end of the article. Extensive experimentation was performed on Portuguese text corpora.*

1. Introduction

Word forms, as atoms, cannot arbitrarily combine with each other. They form new composites by both imposing and satisfying certain requirements. A word uses a linguistic requirement (constraint or preference) in order to restrict the type of words with which it can combine in a particular position. The requirement of a given word is characterized by at least two different objects: the position occupied by the words that can be combined with the given word and the condition that those words must satisfy in order to be in that position. For a word w and a specific description of a location *loc*, the pair $\langle loc, w \rangle$ represents a **position** with regard to w. In addition,

* Departamento de Língua Espanhola, Faculdade de Filologia, Universidade de Santiago de Compostela, Campus Universitario Norte, 15782 Santiago de Compostela, Spain. E-mail: gamallo@fct.unl.pt.
† Faculdade de Informática, Pontifícia Universidade Católica do Rio Grande do Sul, Av. Ipiranga 6681 prédio 30 bloco 4, CEP 90619-900 Porto Alegre (RS), Brazil. E-mail: agustini@inf.pucrs.br.
‡ Department of Computer Science, Faculty of Science and Technology, Universidade Nova de Lisboa, Quinta da Torre, 2829-516, Caparica, Portugal. E-mail: gpl@di.fct.unl.pt.

Submission received: 13th June 2004; Revised submission received: 4th May 2004; Accepted for publication: 17th June 2004

condition *cond* represents the set of linguistic properties that words must satisfy in order to be in position $\langle loc, w \rangle$. So a linguistic requirement of w can be represented as the pair:

$$\langle \langle loc, w \rangle, cond \rangle \quad (1)$$

Consider, for instance, position ⟨*of_right, ratification*⟩, where *of_right* is a location described as *being to the right of preposition of*. This position represents the argument slot *ratification of [_]*. Considering also that *cond* stands for the specific property *being a nominal phrase (np) whose head denotes a legal document* (abbreviated by *doc*), then the pair ⟨⟨*of_right, ratification*⟩, *doc*⟩ means that the particular position *ratification of [_]* selects for nouns denoting legal documents. In other words, *ratification* requires nominal arguments denoting legal documents to appear after preposition *of*. Suppose that there exist some words such as *law*, *treaty*, and *constitution* that are nouns denoting legal documents. Then it follows that they fill the condition imposed by *ratification* in the *of_right* location. An expression like *the ratification of the treaty* is then well-formed, because *treaty* satisfies the required condition.

Let us look now more carefully at several linguistic issues we consider to be important to characterize the notion of linguistic requirement: extensionality/intensionality, soft/hard requirements, the scope of a condition, syntactic/semantic requirements, and corequirements.

A condition can be defined either intentionally or extensionally. For example, the two specific properties *being the head of an np* and *being a legal document* are used to define intensionally the condition imposed by position ⟨*of_right, ratification*⟩. However, it is also possible to define it extensionally by enumerating all those words that actually possess such properties: for example, *law, treaty*, and *constitution.*

Moreover, the process of satisfying a condition can be defined as a binary action producing a Boolean (yes/no) value. From this point of view, a word either satisfies or does not satisfy the condition imposed by another word in a specific location. This is a **hard** requirement. By contrast, the satisfaction process can also be viewed as a **soft** requirement, in which some words are "preferred" without completely excluding other possibilities. In Beale, Niremburg, and Viegas (1998), hard requirements are named **constraints**, whereas the term **preferences** is employed for soft requirements. In the following, we use one of these two terms only if it is necessary to distinguish between hard and soft requirements. Otherwise, **requirement** is taken as the default term.

Let's describe now what we call the **scope** of a condition. A position imposes a specific condition on the words that can appear in that position. Yet a specific condition is generally imposed not by only one position, but by a large set of them. If a condition were bound only to a particular position, every combination of words would be a noncompositional idiomatic expression. So speakers could not combine words easily, and new composite expressions would be difficult to learn. The scope of a condition embraces the positions that use it to restrict word combination. For instance, the condition imposed by *ratification of [_]* seems to be the same as the one imposed by the verb *ratify* on the words appearing at its right: ⟨*right, ratify*⟩ (*to ratify [_]*). In addition, these positions also share the same conditions as *to approve [_], to sign [_],* or *signatories to [_]*. Each of these similar positions is within the scope of a specific condition, namely, *being an np whose head denotes a legal document.* In this article, we assume that every linguistic condition is associated with a set of similar positions. This

set represents the scope of the condition. The larger the set of similar positions, the larger the condition scope, and the more general the property used to characterize the condition.

We distinguish syntactic and semantic requirements. A syntactic requirement is characterized by both a position and a morpho-syntactic condition. For instance, requirement ⟨⟨*of_right, ratification*⟩, *np*⟩ consists of a position, ⟨*of_right, ratification*⟩, which selects for a nominal phrase. Note that the different syntactic requirements of a word can serve to identify the set of subcategorization frames of that word. Note also that, in some cases, a particular position presupposes a particular morpho-syntactic condition. In our example, position ⟨*of_right, ratification*⟩ requires only a *np*. So we can use this position as a shorter form of the syntactic requirement ⟨⟨*of_right, ratification*⟩, *np*⟩. We call a **syntactic position** a position that presupposes a specific morpho-syntactic condition. On the other hand, a **semantic requirement** (also known as **selection restriction**) is characterized by both a position and a semantic condition, which presupposes a syntactic one. So ⟨⟨*of_right, ratification*⟩,*doc*⟩ means that position ⟨*of_right, ratification*⟩ selects for the head of a *np* denoting a legal document. Condition *doc* presupposes then a *np*. Identifying a particular semantic requirement entails the identification of the underlying syntactic one.

The final linguistic issue to be introduced is the phenomenon referred to as **corequirements**. It is assumed that each syntactic dependency between two words (which are the heads of two phrases) is composed of two complementary requirements. For instance, it seems that two different requirements underlie the expression *ratification of the treaty:* ⟨*of_right, ratification*⟩ (*ratification of [_]*) needs to be filled by words like *treaty*, while ⟨*of_left, treaty*⟩ (*[_] of the treaty*) needs to appear with words such as *ratification*.

The main objective of this article is to describe an unsupervised method for learning syntactic and semantic requirements from large text corpora. For instance, our method discovers that the word *secretary* is associated with several syntactic positions (i.e., positions with morpho-syntactic conditions), such as *secretary of [_], [_] of the secretary, [_] to the secretary*, and *[_] with the secretary*. The set of syntactic positions defined by a word can be used to characterize a set of subcategorization frames. The precise characterization of these frames remains, however, beyond the scope of this article. In addition, for each syntactic position, we assess the specific semantic condition a word needs to fill in order to appear in that position. Another important objective of the article is to use the semantic requirements to capture contextually relevant semantic similarities between words. In particular, we assume that two words filling the same semantic requirement share the same contextual word sense. Consequently, learning semantic requirements also leads us to induce word senses. Suppose that the word *organization* fills the condition imposed by *secretary of [_]*. In this syntactic context, the word denotes a social institution and not a temporal process or an abstract setup.

To achieve our objectives, we follow a particular clustering strategy. Syntactic positions (and not words) are compared according to their word distribution. Similar syntactic positions are put in more clusters following some constraints that are defined later. Each cluster of positions represents a semantic condition. The features of each cluster are the words that can fill the common condition imposed by those positions: They are the fillers. They are used to extensionally define the particular condition they can fill. That is, a condition is defined by identifying those words likely to appear in positions considered similar. Given that a condition is extensionally defined by the words that are able to fill it, our method describes the process of

satisfying a condition as a Boolean constraint (yes/no) and not as a probabilistic preference. The similar positions defining a cluster are within the **scope** of a particular semantic condition. The association between each position of the cluster and that condition characterizes the semantic requirement of a word. This learning strategy does not require hand-crafted external resources such as a WordNet-like thesaurus or a machine-readable dictionary.

The information captured by this strategy is useful for two different NLP disambiguation tasks: selecting contextual senses of words (word sense disambiguation) and solving structural ambiguity (attachment resolution). This article is focused on the latter application.

In sum, the main contribution of our work is the large amount of linguistic information we learn for each lexical word. Given a word, we acquire, at least, three types of information: (1) an unordered set of syntactic positions, which is a first approximation to define the set of subcategorization frames of the given word, (2) the semantic requirements the word imposes on its arguments, and (3) the different contextual senses of the word. By contrast, related work focuses only on one or two aspects of this linguistic information. Another contribution is the use of corequirements to characterize the arguments of a word.

To conclude the introduction, let's outline the organization of the article. In the next section, we situate our approach with regard to related work on acquisition of linguistic requirements. Later, in sections 3 and 4, we describe in detail the main linguistic assumptions underlying our approach. Special attention will be paid to both the relativized view on word sense (i.e., contextual sense) and corequirements. Then, section 5 depicts a general overview of our strategy. Two particular aspects of this strategy are analyzed next. More precisely, section 6 describes both how syntactic positions are extracted and how they are clustered in larger classes (section 7). Finally, in section 8, we evaluate the results by measuring their performance in a particular NLP task: syntactic-attachment resolution.

2. Statistics-Based Methods for Learning Linguistic Requirements

During the last years, various stochastic approaches to linguistic requirements acquisition have been proposed (Basili, Pazienza, and Velardi 1992; Hindle and Rooth 1993; Sekine et al. 1992; Grishman and Sterling 1994; Framis 1995; Dagan, Marcus, and Markovitch 1995; Resnik 1997; Dagan, Lee, and Pereira 1998; Marques, Lopes, and Coelho 2000; Ciaramita and Johnson 2000). In general, they follow comparable learning strategies, despite significant differences observed. In this section, we present first the common strategy followed by these approaches, and then we focus on their differences. Special attention is paid to lexical methods. At the end, we situate our strategy with regard to the related work.

2.1 A Common Strategy

The main design of the strategy for automatically learning requirements is to compute the association degree between argument positions and their respective linguistic conditions. For this purpose, the first task is to count the frequency with which $\langle\langle loc, w\rangle, cond\rangle$ occurs in a large corpus:

$$F(\langle\langle loc, w\rangle, cond\rangle) \quad (2)$$

where F counts the frequency of co-occurring $\langle loc, w\rangle$ with *cond*. Then this frequency is used to compute the conditional probability of *cond* given position $\langle loc, w\rangle$:

$$P(cond \mid \langle loc, w\rangle) \tag{3}$$

This probability is then used to measure the strength of statistical association between $\langle loc, w\rangle$ and *cond*. Association measures such as mutual information or log-likelihood are used for measuring the degree of (in)dependence between these two linguistic objects. Intuitively, a high value of the association measure is evidence of the existence of a true requirement (i.e., a type of linguistic dependence).

The stochastic association values obtained by such a strategy turn out to be useful for NLP disambiguation tasks such as attachment resolution in probabilistic parsing and sense disambiguation.

2.2 Specific Aspects of the Common Strategy

Despite the apparent methodological unanimity, approaches to learning requirements propose different definitions for the following objects: association measure, position $\langle loc, w\rangle$, and linguistic condition *cond*. Many approaches differ only in the way in which the association measure is defined. Yet such differences are not discussed in this article.

As regards position $\langle loc, w\rangle$, we distinguish, at least, among three different definitions. First, it can be considered as a mere word sequence (Dagan, Marcus, and Markovitch 1995): For instance, $\langle right, w\rangle$, where *right* means *being to the right of*. Second, a position can also be defined in terms of co-occurrence within a fixed window (Dagan, Lee, and Pereira 1998; Marques, Lopes, and Coelho 2000). Finally, it can be identified as the head or the dependent role within a binary grammatical relationship such as subject, direct object, or modifier (Sekine et al. 1992; Grishman and Sterling 1994; Framis 1995). In section 4, we pay special attention to the grammatical characterization of syntactic positions.

As far as *cond* is concerned, various types of information are used to define a linguistic condition: syntactic, semantic, and lexical information. The approaches to learning requirements are easily distinguished by how they define *cond*. Table 1 displays three different ways of encoding the condition imposed by verb *approve* to the nominal *the law* in the expression *to approve the law*.

Requirement conditions of the pairs in Table 1 represent three descriptive levels for the linguistic information underlying the nominal expression *the law* when it appears to the right of the verb *approve*.[1] The properties *np*, *doc*, and *law* are situated at different levels of abstraction. The morpho-syntactic tag *np* conveys more abstract information than the semantic tag *doc* (document), which, in turn, is more general than the lemma *law*. Some conditions can be inferred from other conditions. For instance, *doc* is used only to tag nouns, which are the heads of nominal phrases. So the semantic tag *doc* entails the syntactic requirement *np*. Likewise, the lemma *law* is associated only with nouns. It entails, then, an *np*.

Some approaches describe linguistic conditions only at the syntactic level (Hindle and Rooth 1993; Marques, Lopes, and Coelho 2000). They count the frequency of pairs

1 In case of Portuguese, for intransitive verbs the occurrence of an *np* to the right of the verb does not mean that the verb is transitive. In fact, this is the standard position of the subject for intransitive verbs.

Table 1
Various levels of encoding linguistic conditions.

Syntactic level	⟨⟨*right, approve*⟩, **np**⟩
Semantic level	⟨⟨*right, approve*⟩, **doc**⟩
Lexical level	⟨⟨*right, approve*⟩, **law**⟩

like ⟨⟨*right, approve*⟩, *np*⟩ in order to calculate the probability of an *np* occurring given ⟨*right, approve*⟩. This probability is then used to compute the degree of association between *approve* and an *np* located to the right. This association value may be useful in different linguistic tasks. For instance, it may serve to solve structural ambiguities (Hindle and Rooth 1993) or to build a subcategorization lexicon (Marques, Lopes, and Coelho 2000). Most approaches to learning syntactic requirements assume that syntactic properties can be identified by means of some specific morphological "cues" appearing in the corpus. For instance, the article *a* following a verb is a clear evidence for an *np* appearing at the *right* of the verb; the preposition *of* following a verb is evidence for an *of_right* complement; and the conjunction *that* after a verb introduces a *that_clause*. Morphological cues are used to easily identify syntactic requirements. This technique allows raw text to be worked on directly. Let us note that these techniques do not allow the acquisition of complete subcategorization frames (Brent 1991; Manning 1993). They are able to acquire that, for instance, *approve* subcategorizes an *np* on two locations: both *right* and *of_right* locations (e.g., *to approve the laws, to approve of the decision*). So they associate that verb with two syntactic arguments. However, they are not able to learn that the two arguments are incompatible and must belong to two different subcategorization frames of the verb. We return to this issue in section 8.1.

In other approaches to requirement learning, linguistic conditions are defined in semantic terms by means of specific tags (Basili, Pazienza, and Velardi 1992; Resnik 1997; Framis 1995). In order to calculate the degree of association between tag *doc* and position ⟨*right, approve*⟩, these approaches count the frequency of pairs like ⟨⟨*right, approve*⟩, *doc*⟩ throughout the corpus. If the association value is higher than that for other related cases, then one might learn that the verb *approve* requires nominal phrases denoting *doc* entities to appear at the right.

According to other learning approaches, the linguistic conditions used to characterize requirements may be situated at the lexical level (Dagan, Lee, and Pereira 1998; Dagan, Marcus, and Markovitch 1995; Grishman and Sterling 1994; Sekine et al. 1992). A pair like ⟨⟨*right, approve*⟩, *law*⟩ matches those expressions containing a form of lemma *law* (e.g., `law, laws, Law, Laws`) appearing to the right of the verb *approve* (to be more precise, to the right of any form of lemma *approve*). The frequency of this pair in the corpus serves to compute the degree of association between *law* and the verb *approve* at the *right*. In these approaches, then, conditions are learned from lexical co-occurrences. From now on, when it is not necessary to distinguish between lemmas and word forms, we use the term "word" for both objects.

To compare the three types of approaches more accurately, let's analyze their behavior regarding different quantitative aspects: (1) the continuum between supervised and unsupervised learning, (2) the continuum between knowledge-poor and knowledge-rich methodology, and (3) the continuum between general- and specific-information acquisition.

2.2.1 Supervised/Unsupervised Learning. The first continuum ranges over the degree of human supervision that is needed to annotate the training corpus. Among the works cited above, Basili, Pazienza, and Velardi (1992) has the highest degree of supervision. This semantic approach requires hand-tagging text nouns using a fixed set of semantic labels. The other approaches involve close to total nonsupervision, since they do not require a training corpus to be annotated by hand. However, some degree of human supervision could be involved in building automatic tools (e.g., a neural tagger in Marques, Lopes, and Coelho [2000]) or linguistic external sources (e.g., WordNet in Resnik [1997]; Framis [1995]; Ciaramita and Johnson [2000]), which are used to annotate the corpus.

2.2.2 Knowledge-Rich/Knowledge-Poor Methods. The second continuum refers to the notions introduced by Grefenstette (1994). He distinguishes the learning methods according to the quantity of linguistic knowledge they require. The most knowledge-rich approaches need a handcrafted thesaurus (WordNet) to semantically annotate nouns of the training corpus (Resnik 1997; Framis 1995; Ciaramita and Johnson 2000). At the opposite end of the continuum, the most knowledge-poor methods are introduced in Dagan, Marcus, and Markovitch (1995) and Dagan, Lee, and Pereira (1998); these merely need to identify lemmas in the corpus.

2.2.3 General/Specific Conditions. As regards the general/specific continuum, **syntactic methods**, that is, approaches to learning syntactic requirements, are the learning methods that use the most general linguistic information. At the opposite end of the continuum, we find the **lexical methods**, that is, those strategies situated at the lexical level. Methods using tags like *doc, human,* and *institution* are situated at an intermediate level and are known as **semantic methods**. One of the most difficult theoretical problems is to choose the appropriate generalization level for learning requirement information.

The syntactic level seems not to be appropriate for solving structural ambiguity. Concerning the parsing task, syntactic information is not always enough to produce a single parse. Consider the following analyses:

$$[_{vp}\text{cut } [_{np}\text{the potato}] [_{pp}\text{with a knife}]] \quad (4)$$

$$[_{vp}\text{cut } [_{np}\text{the potato } [_{pp}\text{with a hole}]]] \quad (5)$$

In order to decide which analysis, either (4) or (5), is correct, we must enlist the aid of our world knowledge concerning cutting actions, use of knives, and the properties of potatoes. In general, we know that knives are used for cutting and that potatoes are objects likely to have holes. So the parser is able to propose a correct analysis only if the lexicon is provided not only with syntactic requirements, but also with information on semantico-pragmatic requirements (i.e., with selection restrictions). Selection restrictions are typically used to capture facts about the world that are generally, but not necessarily, true (Androutsopoulos and Dale 2000). So the main goal of semantic and lexical methods is precisely the acquisition of selection restrictions.

As has been mentioned previously, semantic methods use handcrafted sources of linguistic knowledge such as WordNet. There are several disadvantages associated

with these knowledge-rich approaches: Manually created thesauri contain many words either having rare senses or missing domain-specific meanings. In sum, the level of semantic information provided by handcrafted thesauri is either too specific or too general, and it is usually incomplete. It seems not to be appropriate for most NLP tasks (Grefenstette 1994). By contrast, lexical methods are able to acquire information at the level of detail required by the corpus domain. They are domain-dependent approaches. However, they are very sensitive to the problem of data sparseness.

2.3 Lexical Methods and the Data Sparseness Problem

Most word co-occurrences (for instance, the co-occurrence of *agreement* with *approve* at location *right*) have very small probabilities of occurring in the training corpus. Note that if they were not observed in the corpus, they would have identical probabilities (i.e., probability 0) to those of incorrect co-occurrences such as *cow* appearing to the right of *approve*. This is what is known as the data sparseness problem. To solve this problem, many lexical methods estimate the probabilities of unobserved pairs by taking into account word similarity. Suppose that the pair $\langle\langle right, approve\rangle, agreement\rangle$ is not observed in the training corpus. To obtain an appropriate measure of the association between *agreement* and $\langle right, approve\rangle$, the degree of association between $\langle right, approve\rangle$ and each word most similar to *agreement* is computed. The main criterion for measuring word similarity is comparing the context distribution of words. The total association value for the specific lexical co-occurrence is the average of these association values.

Information on word similarity is used to generalize the pairs appearing in the corpus and to smooth their co-occurrence probabilities. That is, very specific requirements described at the lexical level can be generalized by means of word similarity information.

For instance, the following pair:

$$\langle\langle right, approve\rangle, MOST\_SIM(agreement)\rangle \tag{6}$$

associates the information $MOST\_SIM(agreement)$ with the position $\langle right, approve\rangle$, where $MOST\_SIM(agreement)$ represents the most similar words to *agreement*: for example, *law*, *treaty*, *accordance*, and *conformity*. The use of word similarity allows the probabilities computed at the lexical level to be smoothed (generalized). Computations involving similar words minimize the data sparseness problem to a certain extent. Lexical methods provided with similarity-based generalizations are found in Sekine et al. (1992), Grishman and Sterling (1994), and Dagan, Lee, and Pereira (1998). Later, in section 8.3.4, we use a lexical method with similarity-based generalization to solve syntactic attachments. The results obtained using this method are explicitly compared to those obtained by our clustering strategy.

The methodology for automatically measuring word similarity is often based on Harris's (1985) distributional hypothesis on word meaning. According to this hypothesis, words occurring in similar syntactic contexts (i.e., in similar syntactic positions) are semantically similar. A simple way of implementing this hypothesis is to compute the similarity between words by comparing the *whole* information concerning their context distribution. Allegrini, Montemagni, and Pirrelli (2003) call this strategy the **absolute view** on word similarity. The absolute view leads to the characterization of

word similarity as an intransitive relation (Dagan, Lee, and Pereira 1998). Let us examine expressions (7)–(10), which show that even if *treaty* is similar to *agreement*, and *agreement* is similar to *conformity*, it does not mean that *treaty* is similar to *conformity*:

to approve the agreement/treaty (7)

to ratify the agreement/treaty (8)

we are in agreement/conformity with your proposal (9)

my signature indicates my agreement/conformity to the rules (10)

Intransitivity makes this type of word similarity rather inefficient for identifying contextual word senses. For instance, it does not help show that *agreement* is similar to *treaty* in quite a different way than it is similar to *conformity*. Expressions (7) and (8) introduce the linguistic contexts in which *agreement* denotes a document containing legal information. This word is considered to be semantically similar to *treaty* with regard to the contexts introduced by verbs *approve* and *ratify*. By contrast, (9) and (10) introduce different linguistic contexts. There, *agreement* conveys a different sense: the verbal act of agreeing. In these contexts, it becomes similar to *conformity*. Word similarity methods based on the absolute view seem to be unable to distinguish such contextual meanings. This shortcoming may disrupt the smoothing process defined above. As *conformity* and *accordance* are part of the most similar words to *agreement*, they are involved in the process of computing the degree of association between this word and ⟨*right, approve*⟩. Yet this is counterintuitive, since they are not semantically required by the verb in such a particular position.

2.4 General Properties of Our Method

The objective of this article is to propose a new strategy for learning linguistic requirements. This strategy is designed to overcome the main drawbacks underlying the different approaches introduced above. Our method can be characterized as follows:

- The information it acquires is described at a semantically *appropriate* level of generalization.
- It is defined as a knowledge-poor and unsupervised strategy.

As regards the first characteristic, we consider the method to be semantically appropriate only if the acquired requirements are useful for solving disambiguation problems such as those illustrated above by parses (4) and (5). So our acquisition method is focused on more specific information than that contained in syntactic requirements. Given a word, our aim is to learn not only the syntactic positions in which that word appears, but also the semantico-pragmatic constraints (i.e., what are broadly called selection restrictions associated with each syntactic requirement. Selection restrictions are extracted from position-word co-occurrences. We thus follow a lexical method. However, selection restrictions are defined in accordance with a theory of word sense that is not based on the absolute view on word similarity. We use a more relativized viewpoint on word senses. In sum, we follow a strategy slightly

different from that described in section 2.3. In the next section, we describe our basic assumptions on word sense and word similarity.

Concerning the second characteristic (i.e., knowledge-poor and unsupervised strategy), our method does not rely on external structured sources of lexical information (e.g., WordNet) or on a training corpus built and corrected by hand. Unlike the semantic methods outlined above (in section 2.2), ours attempts to reduce human intervention to a minimum.

3. The Foundations of Our Learning Strategy

In this section, we outline the basic assumptions underlying our learning strategy. This strategy relies on a particular definition of semantic condition (sections 3.1 and 3.2) and a relativized view on word similarity (section 3.3), as well as a specific viewpoint on word sense (section 3.4).

3.1 Extensional Definition

Given a requirement $\langle\langle loc, w\rangle, cond\rangle$, we define a semantic condition, *cond*, as the set of words that can occur in position $\langle loc, w\rangle$. This means that linguistic conditions are defined extensionally. For instance, consider again position $\langle right, approve\rangle$ and one of its possible conditions, namely, *doc*, which, as has been shown, means *being a noun denoting a legal document*. This condition is extensionally defined by enumerating the set of words likely to occur with both $\langle right, approve\rangle$ and their *similar* positions. Identifying such a word set is not a trivial task. This set is not a closed, fixed, and predefined list of nouns. Rather, it turns out to be a set open to a great variety of extensions, since it can be modified as time, domain, or speaker change. The aim of our method is to learn, for each argument position, the open set (or sets) of words it requires. Each word set represents, in extensional terms, a specific linguistic condition. For this purpose, we opt for the following learning strategy.

The condition imposed by an argument position is represented by the set of words actually appearing in this position in the training corpus. For instance, let's suppose that $\langle right, approve\rangle$ occurs with four different words: *law, agreement, convention,* and oil (to simplify the explanation, frequencies are not taken into account). For the present, we know only that these words are mere candidates to satisfy the condition imposed by that position. In order to actually know whether or not the candidate fillers satisfy such a condition, we select the most *similar* positions to $\langle right, approve\rangle$. So we get clusters of similar positions imposing the same condition. Consider, for instance, the following cluster:

$$\{\langle right, approve\rangle, \langle right, ratify\rangle, \langle to\_right, signatories\rangle, \langle by\_left, becertified\rangle, \langle of\_right, ratification\rangle\} \quad (11)$$

which is made of positions sharing features such as

$$law, agreement, article, treaty, convention, document \quad (12)$$

So, cluster features in (12) are the words that may fill the specific condition imposed by the similar positions in (11). These words can be viewed as fillers satisfying the

intensional property *being a noun denoting a legal document*. Note that (12) contains some words (e.g., *article* and *treaty*) that do not actually occur with position ⟨*right*, *approve*⟩ in the corpus. However, as these words actually occur with most of the positions that are similar to ⟨*right*, *approve*⟩, we may assume that they satisfy the condition of this particular position. This is the technique we use to generalize (smooth) occurrences of position-word pairs that are not observed in the training corpus. Details of our method of clustering are given in section 7.2. Notice also that the set of fillers does not include the word *oil*. This word does not belong to the set of shared features because it does not occur with any of the positions similar to ⟨*right*, *approve*⟩. This is the method we use to identify and remove invalid associations between a position and a word. It is explained in section 7.1.

In sum, positions are considered to be similar to one another because they impose the same condition (i.e., they share the same selection restrictions). As has been noted earlier, similar positions are within the scope of one common requirement. The set of similar positions in (11) represents the scope of condition (12). The fillers are those words that characterize the extension of such a condition.

3.2 Hard Requirements

We assume that the process of condition satisfaction may be defined as a Boolean function and not as a probabilistic one. The value of the association between, for instance, the word *treaty* and the position ⟨*right*, *approve*⟩ is either yes or no. Our method merely attempts to learn whether or not there is a true association between them. If the association is actually true, then we learn that the word satisfies the condition. Hard requirements can easily be used to constrain the grammar of a symbolic parser. In particular, we use them to improve the parser described in Rocio, de la Clergerie, and Lopes (2001). Although linguistic constraints are defined in Boolean terms, they are open to potential changes. Clusters and their features are supposed to be modified and extended as the training corpus grows and is progressively annotated with more trustworthy syntactic information. Moreover, a new domain-specific corpus can lead us not only to create new clusters, but also to tune old ones. From this viewpoint, Boolean constraints cannot be considered necessary and sufficient conditions. They evolve progressively.

3.3 Relativized Word Similarity

Our learning strategy relies on a specific assumption on word similarity. We are interested in computing similarity between words with regard to a set of similar positions. So we must first compute similarity between positions. As has been mentioned before, similar positions impose the same linguistic condition. Hence, they are likely to be filled by the same set of words. Statistically, this means that they have **similar** word distribution. A definition of this similarity is given in section 7.1. Unlike in the absolute view stated above, we are not interested in measuring similarity between words on the basis of the distribution of all their corpus-based positions (their whole context distribution). Our aim is, first, to compute the similarity between positions via their word distribution. Positions are in fact less ambiguous than words. Then, we consider two words to be similar if they occur with at least a pair of similar positions. This way of using similar positions allows all possible dimensions of similarity of a given word to be captured. This is close to the "relativized view" on word similarity offered by Allegrini, Montemagni, and Pirrelli (2003).

3.4 Contextual Hypothesis on Word Sense

Behind this account of similarity, there is a particular view of word sense that is not far from that of Schütze (1998):

> *Contextual Hypothesis for Word Sense*: A set of similar positions defines a particular type of context. A word occurring with positions of the same type keeps the same sense. The sense of a word changes if the word appears in a different type of context.

For instance **agreement** refers to a legal document when it satisfies the requirement of similar positions such as *to approve [_]* or *ratification of [_]*. By contrast, this word denotes a verbal act when it appears in positions such as *in [_] with your proposal* or *[_] to the rules.*

According to this hypothesis, identifying word senses relies on identifying sets of similar positions (i.e., types of contexts). The noun *book,* for instance, can denote at least three different contextual senses provided it appears in three context types: for example, physical actions (carrying the book, putting it on the table, etc.), symbolic processes (writing or reading books), and economic transactions (selling or buying books). This notion of word sense is dependent on the ability to grasp classes of contexts, that is, the ability to learn clusters of similar positions. The more accurate is this ability, the more precise are the senses identified in a particular corpus. This means that the set of senses associated with a word cannot be predefined by an external lexical resource like WordNet or any machine-readable dictionary. Word senses are dynamically learned as the text is processed and positions are organized in semantically homogenous clusters. Each cluster of similar positions (or context type) represents a particular word sense. From this viewpoint, the set of contextual senses of a word represents its whole meaning. Such a notion of word meaning is in accordance with the encyclopedic hypothesis on lexical meaning within the cognitive grammar framework (Langacker 1991). According to this hypothesis, every word or lexical unit is associated with a **continuum** of encyclopedic knowledge (the word meaning). The use of the word in a particular context makes a partial aspect of this continuum more salient (a specific word sense).

Within a particular corpus, we assume that the **meaning** of a word is defined by the context types that organize the different positions of the word. In other words, a word's meaning is described by identifying the types of requirements the word fulfills. In the next section, we explore the notions of requirement and syntactic position.

4. Syntactic Positions and Corequirements

This section discusses the general properties of syntactic positions and their role in extracting linguistic requirements. Syntactic positions are defined here as internal elements of binary dependencies. Two aspects of dependencies are retained: the head-dependent distinction and the predicate-argument structure. Special attention is paid to corequirements.

4.1 Head-Dependent Distinction

The **head-dependent pattern** takes over the process of transferring morpho-syntactic features to higher grammatical levels. A composite expression inherits the features of

Table 2
Two binary dependencies and their positions.

Dependencies	Contexts
(*robj*; *ratify*$^{\downarrow}$, *law*$^{\uparrow}$)	⟨*robj_down*, *ratify*⟩
	⟨*robj_up*, *law*⟩
(*mod*; *dinner*$^{\downarrow}$, *long*$^{\uparrow}$)	⟨*mod_down*, *dinner*⟩
	⟨*mod_up*, *long*⟩

the headword. There are two different locations (or grammatical roles) within a binary dependency: the **head** and the **dependent**. Consider the binary dependencies shown in the first column of Table 2, which represent the expressions *to ratify the law* and *long dinner.* The grammatical relations between the two words are expressed by both *robj*, which stands for the nominal object appearing to the right of the verb,[2] and *mod*, which stands for the noun-adjective dependency. The word indexed by $^{\downarrow}$ (**down** location) plays the role of head, whereas the word indexed by $^{\uparrow}$ (**up** location) plays the role of dependent. Since a binary dependency is constituted by two grammatical locations, we can split the dependency into two complementary syntactic positions.

Each pair of positions in the second column of Table 2 was extracted from a binary dependency. We show below that the two positions extracted from a dependency are associated with specific semantic conditions. Hence, they can be used to characterize syntactico-semantic requirements. In our work, the different types of binary relations from which we extract all positions are *lobj, robj, iobj_prepname*, *aobj_prepname, prepname,* and *mod*. Relation *lobj* designates the nominal object appearing to the left of verb, *robj* represents the nominal object appearing to the right of the verb, *iobj_prepname* introduces a nominal after a verb and a preposition, *aobj_prepname* represents a nominal after an adjective and a preposition, *prepname* corresponds to a nominal following a noun and a preposition, and *mod* refers to the adjective modification of nouns. Note that each relation conveys not only two argument positions, but also specific morpho-syntactic conditions. *robj,* for instance, signals that there is an *np* to the right of a *vp*. So ⟨*robj_down, ratify*⟩ contains the same information as the syntactic requirement ⟨⟨*robj_down, ratify*⟩, *np*⟩, while ⟨*robj_up, law*⟩ is equivalent to ⟨⟨*robj_up*, *law*⟩, *vp*⟩.

4.2 Predicate-Argument Structure

Besides the head-dependent pattern, binary dependencies are also organized as predicative structures: **Predicate**(Argument). While the former pattern drives the process of inheriting morpho-syntactic information throughout grammatical levels, the latter is directly related to semantic requirements. This section starts by introducing the standard account concerning the role of the **Predicate**(Argument) structure in the process of imposing linguistic requirements. Then we make new assumptions about what we consider to be a more accurate notion of requirement information. This notion is modeled by means of what we call corequirements. Corequirements are used later, in sections 6 and 7, to elaborate our learning method.

2 In Portuguese, a right object (without governing preposition) can be elaborated, under specific conditions, as either a direct object or a subject.

4.2.1 Simple Requirements. It is broadly assumed that a binary syntactic dependency is constituted by both the word that imposes linguistic constraints (the predicate) and the word that must fill such constraints (its argument). In a syntactic dependency, each word is considered to play a fixed role. The argument is perceived as the word specifying or modifying the syntactico-semantic constraints imposed by the predicate, while the predicate is viewed as the word that is specified or modified by the former. Notice that a **predicate** is not defined here in model-theoretic terms. We inherit the intuitive definition assumed in the linguistic tradition of dependency grammar (Hudson 2003). According to this tradition, a predicate is the semantic representation of one of the two words taking part in a binary dependency. More precisely, it is the representation of one word (either head or dependent) that actually imposes semantic requirements on the other word.

In standard linguistic approaches, the **Predicate**(Argument) structure is the semantic counterpart of the head-dependent pattern. The former relates to the latter in the following way. Typically, the dependent word playing the role of Argument is conceived as the **complement** or **object** of the head (see Figure 1). By contrast, when it plays a more active role, behaving more like a Predicate, it is viewed as a **modifier** or the **adjunct** of the head (Figure 2). In other words, the dependent of a head-dependent structure is described either as a passive complement, if it satisfies the linguistic requirements of the head (Argument role), or as an active modifier, when the dependent itself requires a specific type of head (Predicative role).

The most typical case of a head being a predicate is when a verb is the head within a direct-object dependency. The noun is viewed here as a complement, that is, as a dependent expression fulfilling the conditions imposed by the verbal predicate. The most typical case of a dependent taken as a predicate is the standard use of an adjective or an adverb. In this case, it is the adjective (or adverb) that imposes the selection restrictions on the noun (or verb), which is the head of the dependency.

By contrast, in case of dependencies such as prepositional relations, it is not possible to distinguish between a complement and a modifier, unless we have access to the specific semantico-pragmatic information conveyed by words. However, there are many cases in which the borderline between complement and modifier is not clear. In these cases, even semantico-pragmatic knowledge is not enough to enable a decision to be made in favor of one particular predicative structure. For instance, consider the expression *to play with a doll*. Which is the word that can be taken as the predicate, and which as the argument?

Linguists have made a considerable effort to discriminate between complements and modifiers (= adjuncts). The complement/modifier distinction is probably one of the most unclear issues in linguistics (Langacker 1991). No linguistic proposal is able to distinguish in absolute terms complements from external adjuncts; for example, in the previous expression, is *with a doll* an internal complement or an adverbial modifier of *play*? In other words, is position ⟨*iobj_with_down,play*⟩ one that requires as argument the noun *doll* (complement construction)? Or, on the contrary, is position ⟨*iobj_with_up, doll*⟩ one that requires as argument the verb *play* (modifier structure)? There is no reliable evidence on which to choose between the two possible requirement structures. Most

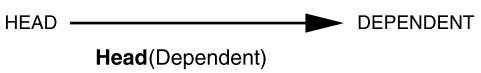

Figure 1
Complement structure.

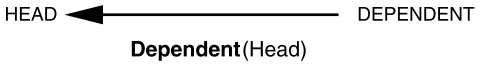

Figure 2
Modifier structure.

linguistic proposals may be situated in one of two general trends: (1) Some linguists interpolate finer distinctions between the two extremes (Pustejovsky 1995). So between true or basic complements and completely optional adjuncts, it is possible to find default complements, shadow complements, and so on which share properties of both complements and adjuncts. (2) A more radical view is to consider the complement/modifier opposition as a continuum in which it is not easy to fix borderlines between what is entirely optional and what is obligatory (Langacker 1991).

The idea of a continuum entails that complements and modifiers cannot be defined in absolute terms. All binary dependencies always contain a certain degree of both complementarization and modification. That is, given a dependency, the head requires the dependent (complementarization), and conversely, the dependent requires the head (modification). We assume in this article that such corequirements underlie any binary dependency.

4.2.2 Corequirements. Recent linguistic research assumes that two words related by a syntactic dependency are mutually constrained (Pustejovsky 1995; Gamallo 2003). Each word imposes linguistic requirements on the other. There does not exist a single, pre-fixed predicate-argument pattern. Each related word is at the same time both a predicate and an argument. We call such a phenomenon corequirement structure.

Consider again the expression *potato with a hole*. It does not seem obvious whether *hole* is the complement or the modifier of *potato* within the *with* dependency. If it is considered the complement, then it is the head *potato* that should be provided with the appropriate requirements. Otherwise, it should be the modifier *hole,* the word imposing specific requirements on the head. Following recent research, we claim, however, that such a radical opposition is not useful for describing linguistic requirements. It is assumed here that two syntactically related expressions presuppose two complementary requirements. In other words, every binary dependency is constituted by two compatible predicate-argument structures.

On the one hand, the noun *potato* requires words denoting parts or properties of potatoes to appear in the *with_down* location. The noun *hole* satisfies such a requirement. On the other hand, the noun *hole* is also provided with selective requirements in the *with_up* location. Indeed, in this location, it requires nouns denoting material objects that can have holes. The noun *potato* fulfills such a condition. Note that the expressions *cut with* a *knife* and *play with* a *doll* could also be considered borderline cases.

Corequirements are not useful only for modeling borderline cases. We believe that they are also pertinent for typical complement structures (e.g., the direct-object relation between verbs and nouns), as well as for typical modifier constructions (i.e., adjective-noun and verb-adverb dependencies). In *long dinner*, for instance, the noun seems to behave as a predicate constraining the adjective to denote a temporal dimension (and not a spatial one). So not only does the adjective disambiguate the noun, but the noun also disambiguates the adjective.

Therefore, according to the assumption on corequirements, two syntactically dependent expressions are no longer interpreted as a standard predicate-argument pair, in which the predicate is the active function imposing semantic conditions on a passive

argument, which matches these conditions. On the contrary, each word of a binary dependency is perceived simultaneously as both a predicate and an argument. That is, each word both imposes semantic requirements and matches semantic requirements in return. Figure 3 depicts a standard syntactic dependency between two words, the head and the modifier, with two **Predicate**(Argument) structures. Figure 4 illustrates the two specific **Predicate**(Argument) structures extracted from the modifier relation between the noun *dinner* (the head) and the adjective *long* (the dependent).

The learning strategy described in the remainder of the article takes advantage of the corequirement structure.

5. System Overview

To evaluate the hypotheses presented above, a software package was developed to support the automatic acquisition of syntactic and semantic requirements. The system is constituted by six main processes, which are displayed as rectangles with solid lines in Figure 5. They are organized as a linear sequence of data transformations. In Figure 5, solid ellipses are used to display the data transformed by these processes. Two local subprocesses (dotted rectangles) build extra data (dotted ellipses), in order to constrain some of the main transformation processes. In the remainder of this section, we merely outline the overall functionalities of these processes. Then in subsequent sections, we describe them in detail.

Tagging and Chunking: Raw text is tagged (Marques and Lopes 2001) and then analyzed in chunks using some potentialities of the shallow parser introduced in Rocio, de la Clergerie, and Lopes (2001). This parser is implemented using tabulation capabilities of the DyALog system (de la Clergerie 2002). The output is a sequence of basic chunks. For instance, the sentence *The President sent the document to the Minister* is analyzed as a sequence of four basic chunks: *np*, *vp*, *np*, and *pp*. These chunks contain neither dependencies nor recursivity.

Attachment Heuristic RA: An attachment heuristic based on right association (RA) is applied to chunk sequences in order to combine pairs of chunks. The headwords of two related chunks form a **syntactic dependency.** Section 6.1 describes some properties of the dependencies extracted using the RA strategy.

Extractor of Position Vectors: Dependencies are used to extract **syntactic positions,** which are internally characterized as vectors of word frequencies. This process is described in section 6.2.

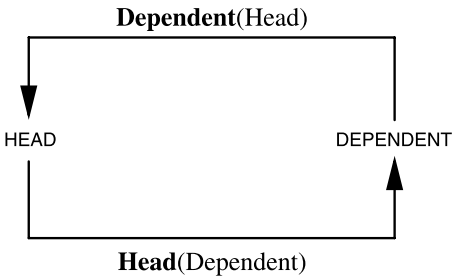

Figure 3
Dependency with corequirements.

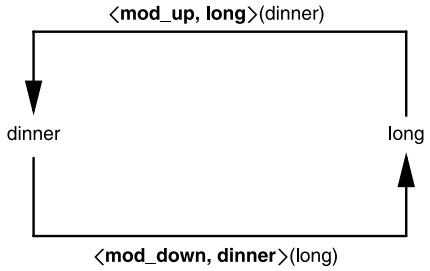

Figure 4
Corequirements in *long dinner*.

Clustering 1: Position vectors are compared with one another using a specific similarity measure. Pairs of similar positions are put in **basic clusters.** A basic cluster is constituted by two similar positions whose features are the words they share. Section 7.1 describes this process.

Clustering 2: Basic clusters are successively aggregated using a conceptual clustering methodology to induce more-general classes of positions. A corpus-based thesaurus, built on the basis of the extracted dependencies, is used to constrain cluster aggregation. We present this process (together with the thesaurus design subprocess) in section 7.2.

Attachment Heuristic CR: Finally, the resulting clusters are used to parse again the chunks and propose new dependencies (section 8). This is accomplished in two steps. First, a lexicon builder module organizes the information underlying the

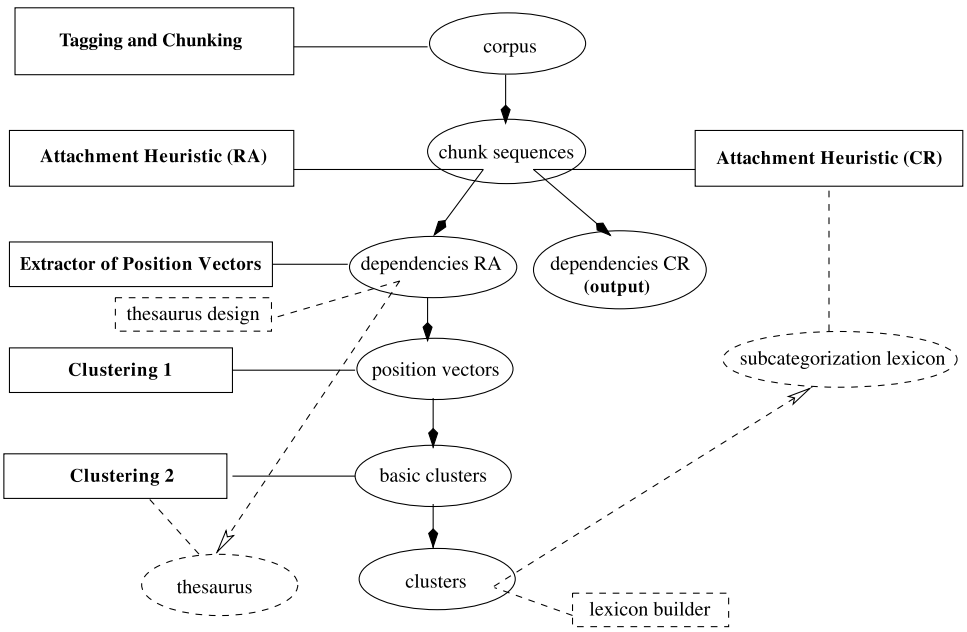

Figure 5
System modules.

learned clusters and builds a lexicon with syntactico-semantic corequirements (see section 8.1). Then, the grammar underlying the parser is provided with a specific attachment heuristic that uses corequirement (CR) information from the lexicon. This heuristic allows the parser to propose a new set of dependencies (section 8.2). We evaluate the resulting dependencies in section 8.3.

The system was implemented on two different Portuguese text corpora: PGR[3] and EC.[4] The experiments that were conducted are described and some results given in section 7.3.

6. Extracting Dependencies and Positions

In this section, we describe two modules of the method: the attachment heuristic RA and the extractor of position vectors. These modules involve the extraction of candidate binary dependencies and syntactic positions.

6.1 Attachment Heuristic RA

Attachment heuristic RA takes as input parses constituted by sequences of chunks. It uses the right-association strategy. That is, a new chunk is preferentially attached to the preceding chunk. The headwords of two attached chunks form a possible binary dependency. Consider the expression

... a lei citada em o anterior parecer... (*the law cited in the previous opinion*) (13)

The RA heuristic allows us to identify three candidate dependencies, which are illustrated in the left column of Table 3. These dependencies are not considered at this point to be actual attachments, but only potential candidates. Later, the parser will be provided with the learned requirements stored in the lexicon, in order to propose new dependencies, which will be the output of the parsing strategy. Note that *lobj* denotes a nominal object appearing to the left of the verb. This cannot be identified with the subject grammatical function. The order of verbal objects is not the main feature by means of which to identify the subject and direct-object functions in Portuguese (and in most Latin languages). The position of verb complements is quite free in these languages. We consider then that grammatical functions are semantic-dependent objects, since we need semantico-pragmatic knowledge to identify them.

In addition, we also provide some dependencies with specific morpho-syntactic information. For instance, verb citar (*to cite*) is annotated using the past participle (*vpp*) tag. This morpho-syntactic information is relevant for defining the semantic requirements of dependencies. As we show later, only semantic information enables us to consider the dependency underlying a lei citada (*the law that was cited*) as being semantically similar to the one underlying citar a lei (*to cite the law*). These dependencies are not merely merged into one single relation using morpho-syntactic rules. Such rules pose some important problems: First, they require specific knowledge

3 PGR (Portuguese General Attorney Opinions) consists of case law documents.

4 EC (European Commission) contains documents on different aspects (legislation in force, social policy, environment, etc.) of the European Commission. This corpus is available at http://europa.eu.int/eur-lex/pt/index.html.

Table 3
Binary dependencies and syntactic positions extracted from expression (13).

Binary dependencies	Syntactic positions
(*lobj*; *citar* : *vpp*$^{\downarrow}$, *lei*$^{\uparrow}$)	⟨*lobj_down, citar : vpp*⟩
(*be cited*, *law*)	⟨*lobj_up, lei*⟩
(*iobj_em*; *citar* : *vpp*$^{\downarrow}$, *parecer*$^{\uparrow}$)	⟨*iobj_em_down, citar : vpp*⟩
(*be cited in report*)	⟨*iobj_em_up, parecer*⟩
(*mod*; *parecer*$^{\downarrow}$, *anterior* : *pre*$^{\uparrow}$)	⟨*mod_down, parecer*⟩
(*opinion, previous*)	⟨*mod_up, anterior : pre*⟩

on particular languages; and second, they introduce a great amount of noise. In our approach, these two dependencies are merged in one cluster only if our learning process provides us with semantic evidence to justify such merging. In fact, one of the objectives of our learning method is to use information on semantic requirements for identifying morpho-syntactic alternations of dependencies: for example, `citada pelo ministro/o ministro citou` (*cited by the Minister/the Minister cited*); `mencionar a lei/mencionou-se a lei` (*to mention the law/the law was mentioned*); `ratificar a lei/ratificação da lei` (*to ratify the law/ratification of the law*). If two morpho-syntactic alternations are considered to share the same semantic requirements, then they will automatically occur in the same cluster. This strategy allows us to reduce language-dependent knowledge to a minimum.

It is also worth noticing that tag *pre* in Table 3 is used to annotate adjectives in the left position with regard to the modified noun (i.e., in the *mod* relation). We distinguish three different adjective relations: the left modifier, the right modifier, and the prepositional object. It is assumed here that these three dependencies can stress different semantic aspects of an adjective. For instance, our strategy led us to learn that `anterior` (*previous*) is semantically similar to `primeiro` (*first*) and `seguinte` (*following*) when it takes the role of left modifier. However, when the adjective is to the right of a noun and is followed by a prepositional object (`anterior a`, *previous to*), it is clustered together with `inferior` (*inferior*) and `igual` (*equal*), which also appear within prepositional dependencies: *equal to*, *inferior to*.

Since the right-association strategy is knowledge-poor, the attachments it proposes give rise to a substantial amount of odd syntactic dependencies (25%), including those caused by POS-tagging errors. The overall precision of the tagger is 96.2%. Yet considering only those tags we use in the learning strategy (i.e., nouns, adjectives, articles, verbs, etc.), the precision is close to 90%. To overcome such a noisy input, we need a reliable learning method.

6.2 Position Vectors

Given that each dependency contains two complementary grammatical locations (head and dependent), we split dependencies into two syntactic positions: the position associated with the head (or down) location and the one associated with the dependent (or up) location. The positions extracted from expression (13) are illustrated in the right column of Table 3. Following the assumption on corequirements, each position must be provided with a particular linguistic requirement.

We represent each syntactic position as a feature vector. Each feature corresponds to a word occurring in the position. The value of the feature is the frequency of the

Table 4
Two position vectors.

⟨*iobj_em_down, citar : vpp*⟩	(nota 53) (parecer 7) (conclusão 3) (informação 2) (regulamento 1) (artigo 1) (*apoio 1*) (*sentido 1*)
(*be cited in [_]*)	*note, report, conclusion, information, regulation, article, support, sense*
⟨*iobj_em_up, parecer*⟩	(afirmar:vpp 9) (defender:vpp 7) (citar:vpp 7) (analisar:vpp 7) (escrever 3) (reafirmar:vpp 2) (esclarecer:vpp 1) (notar:vpp 1) (publicar:vpp 1) (concluir:vpp 1) (assinalar:vpp 1)
([_] *in the report*)	*be affirmed, be defended, be cited, be analyzed, writer, be affirmed again, be clarified, be noted, be published, be concluded, be pointed out*

word in that position. A position is thus defined by means of its word distribution. As has been mentioned before, those words appearing in a particular position can be used to represent, in extensional terms, a first approximation of the semantic condition the position requires (i.e., its selection restrictions). Clustering enables us to enlarge the scope of each condition. In Table 4, we illustrate the word distribution of the two complementary positions underlying citada no parecer (*be cited in the report*).

Note that those words occurring once in a position are also considered as features. This allows us to minimize the data sparseness problem. Linguistic corpora are sparse in the sense that most co-occurrences occur few times in a given corpus. So, if co-occurences with lower frequencies were not used by the learning strategy, pertinent linguistic information would be missing, and coverage would remain low. In order to minimize missing information and coverage reduction, we retain infrequent words in position vectors.

Nevertheless, taking into account infrequent co-occurrences increases noise and thus may disturb the learning task. There are a number of noise sources: words missing from the dictionary, words incorrectly tagged, wrong attachments, etc. The position shown in the first line of Table 4 occurs with at least two words that are not syntactically required: apoio (*support*) and sentido (*sense*).[5] Note that these words have frequency 1 in that position. Retaining requirements with frequency 1 enables us to retain other words that are syntactically and semantically appropriate for that position, such as artigo (*article*) and regulamento (*regulation*), which also occur only once. The next step of our method (Clustering 1) focuses on the automatic removal of odd features introduced in position vectors.

7. Clustering of Positions

Positions that share similar features are combined into clusters. Clustering is divided into two different agglomerative processes: Clustering 1 and Clustering 2.

7.1 Clustering 1

Clustering 1 builds pairs of similar positions called basic clusters. A basic cluster is the result of merging two positions considered to be similar. The features associated with a basic cluster are only those words appearing in both similar positions. This allows us

5 Word sentido (*sense*) appears in that position, not as a verb complement, but as a member of the prepositional locution no sentido de (*in the sense that*), which is attached to the whole sentence.

to filter out odd features from clusters. Features defining a basic cluster are, then, the most reliable fillers of the semantic condition imposed by the two similar positions. Those words that are not required by both positions are removed from the cluster. The algorithm of this process is the following:

Similarity: We calculate the **similarity** between each pair of positions. To do this, we measure the distance between their word distributions (see the details of this measure below).

Selection: Then, for each position, we select the N (where $N = 20$) most similar ones.

Aggregation: Then, given a position and the list of N most similar positions, we merge the position with each member of the list. So, given a position, we create N aggregations of that position with one similar position.

Filtering: Finally, for each aggregation of two similar positions, we select the intersection of their features; that is, the features of a basic cluster are those words that appear in both positions.

As a result of this process, we obtain a set of basic clusters, each augmented by reliable features. The aim is to automatically filter out noisy features from each pair of similar syntactic positions. Many incorrectly tagged words are removed at the filtering step.

Let's take an example. Consider the position shown in the first row of Table 4, that is, ⟨*iobj_em_down, citar : vpp*⟩. According to our similarity measure, its word distribution is similar to that of the following positions:[6]

$$\begin{array}{ll} \langle \textit{iobj\_em\_down, mencionar : vpp} \rangle & \langle \textit{iobj\_em\_down, cite} \rangle \\ \langle \textit{iobj\_em\_down, assinalar : vpp} \rangle & \langle \textit{de\_down, leitura} \rangle \\ \langle \textit{iobj\_em\_down, referir : vpp} \rangle & \langle \textit{iobj\_em\_down, referenciar : vpp} \rangle ... \end{array} \quad (14)$$

Then, ⟨*iobj_em_down, citar : vpp*⟩ is merged with each one of the above positions. Note that this position is similar to the position associated with the active form, `citar`. Finally, each pair of similar positions (i.e., each basic cluster) is defined by the words they have in common. For instance, take the basic cluster shown in (15):

$$\{\langle \textit{iobj\_em\_down, citar : vpp} \rangle + \langle \textit{iobj\_em\_down, mencionar : vpp} \rangle\} = \\ \texttt{nota conclusão informação artigo} \\ (\textit{note, conclusion, information, article}) \quad (15)$$

Looking at those words appearing as prepositional objects of both *cited in [_]* and *mentioned in [_]*, one can see that they are semantically homogeneous. The filtered features no longer include odd words such as *support* and *sense* (see Table 4). Indeed,

6 English translation of (14): *mencionar = be mentioned in [_], cite = cite in [_], assinalar = be pointed out in [_], leitura = reading of [_], referir = be referred to in [_], referenciar = be made reference to in [_].*

the process of selecting the words shared by two similar positions relies on the contextual hypothesis stated above in section 3.4, as well as on the following corpus-based observation: Those words whose appearance in a particular position would be incorrect are not likely to occur in similar positions.

Merging two similar positions by intersecting their features allows a semantic condition to be associated with two positions. In (15), a single set of words is associated with the two positions, since the positions have in common the same semantic condition (or selection restrictions). However, the scope of the condition is still too narrow: It merely embraces two positions. In order to extend the scope of semantic conditions, we cluster them using a less restrictive clustering process that allows us to build more general classes of words and positions.

Before explaining the following process (Clustering 2), let us describe the measure used to calculate the similarity between syntactic positions. We use a particular weighted version of the Lin (1998) coefficient. Our version, however, does not use **pointwise mutual information** to characterize the weight on position-word pairs. As Manning and Schütze (1999) argued, this does not seem to be a good measure of the strength of association between a word and a local position. When the similarity between two positions is computed using our method, higher scores are assigned to rare attributes (i.e., words in our case) of compared objects (positions). By contrast, the pointwise mutual information measure is not sensitive to the fact that frequent pairs can have a strong association. In order to resolve this problem, we use a weight very similar to that proposed in Grefenstette (1994). Consequently, we employ, on the one hand, the general structure of the Lin coefficient, and on the other, the weight proposed by Grefenstette.

Words are weighted considering their dispersion (global weight) and their conditional probability given a position (local weight). The weight *Assoc*, measuring the degree of association between word w and position p, is computed by equation (16):

$$Assoc(p,w) = \log_2(P_{MLE}(w|p)) * \log_2(disp(w)) \tag{16}$$

On the other hand, the conditional probability P_{MLE} is estimated by using the **maximum likelihood estimate** (MLE), which is calculated in (17):

$$P_{MLE}(w|p) = \frac{f(p,w)}{F(p)} \tag{17}$$

where $f(p,w)$ represents the frequency of word w appearing in position p, and $F(p)$ is defined, for a particular position, as the total sum of its word frequencies: $\sum_i f(p,w_i)$. On the other hand, word dispersion, *disp*, is defined as the following mean:

$$disp(w) = \frac{F(w)}{\text{number of positions } f \text{ or } w} \tag{18}$$

where $F(w)$ is defined as the total sum of position frequencies of w: $\sum_i f(p_i,w)$. Higher values are assigned by equation (18) to those words that are not dispersed, that is, to

those words frequently appearing in few positions. *disp* measures the ability of a word to be semantically selective with regard to its positions. So the Lin coefficient (*LIN*) between two positions is computed using equation (19):

$$LIN(p_1, p_2) = \frac{\sum_{\{w:\exists(p_1,w),\exists(p_2,w)\}} (Assoc(p_1, w) + Assoc(p_2, w))}{\sum_{\{w:\exists(p_1,w)\}} Assoc(p_1, w) + \sum_{\{w:\exists(p_2,w)\}} Assoc(p_2, w)} \quad (19)$$

In the numerator of (19), the condition of the summation indicates that each word w must be found with both positions p_1 and p_2. In the denominator, w varies over all words found in p_1 and p_2.

7.2 Clustering 2

Basic clusters are the input objects of the second process of clustering. We use an agglomerative (bottom-up) clustering for aggregating basic clusters into larger ones. The clustering algorithm is described in Table 5. According to this algorithm, two objects are clustered if they satisfy the following restrictions: (1) they have the same number of features (i.e., words), (2) they share more than 80% common features, and (3) the features that are different must be thesaurically related to at least one of the common features. In order to provide words with thesaurical relations, we automatically build a thesaurus of similar words. Details of the thesaurus design are given in section 7.5.

Figure 6 shows how two basic clusters are merged into one more general class of positions. For two basic clusters such as CL_00013, which contains the features *note*, *article*, *dispatch*, *document*, and *text*, and CL_03202, whose features are *article*, *dispatch*, *document*, *text*, and *opinion*, we obtain the more general cluster CL_04447, which is constituted by all the different positions and words of its basic components. Note that the two basic clusters are different with regard to two features: *note* and *opinion*. According to our clustering restrictions, the two clusters can be merged if each different feature (i.e., *note* and *opinion*) is thesaurically related to at least one of the common features: *article*, *dispatch*, *document*, and *text*. A word is thesaurically related to another if it belongs to the list of most similar words, a list that was automatically generated and entered in our thesaurus. The thesaurus, then, is used to control and constrain the construction of abstract classes of positions. In addition, the larger class, CL_04447, allows us to induce collocation data that does not appear in the corpus. For instance, we induce that the word `parecer` (*opinion*) may appear in position ⟨*iobj_em, mencionar : vpp*⟩ (*mentioned in [_]*). Similarly, we also learn that word `nota` (*note*) can occur with ⟨*iobj_em, referenciar : vpp*⟩ (*made reference to in [_]*).

7.3 Tests and Results

We tested our learning strategy over two training corpora, PGR and EC.[7] Data concerning the information extracted from these two corpora are presented in Table 6.

The clusters generated by Clustering 2 are used to build a lexicon of words along with their syntactic and semantic requirements. Each corpus has its own lexicon. Later, in section 8.1, we describe how this information is stored in the lexicon entries.

7 Some results can be consulted at http://di165.di.fct.unl.pt/~agustini/restr_web.

Table 5
Algorithm of Clustering 2.

Input	Set of basic clusters organized by number of features.
Output	A list of larger clusters representing classes of semantic conditions.
Step 1	*Prerestrictions on candidates to be clustered* For each *obj*, select those objects that • have the same number of features than *obj* AND • share at least 80% of features
Step 2	*Similarity restrictions* From candidates extracted in step 1, take those objects • that share all features with *obj* OR • the different features of which are related by a thesaurus
Step 3	*Merging objects and their features* *obj* is merged with all objects filling the conditions stated in steps 1 and 2. The new object has the following properties: • It is constituted by the union of the features defining the merged objects. • It is put together with objects having the same number of features.
Iteration	Repeat steps 1, 2, and 3, increasing the number of features, until no cluster fulfills the restrictions.

Learned clusters represent linguistic requirements that cannot be reduced to a smaller set of general syntactico-semantic roles, such as Agent, Patient, Theme, and Instrument. On the other hand, they cannot be associated with word-specific roles like, for instance, Reader, Eater, and Singer. The level of elaboration of these clusters ranges from very abstract to very specific lexical levels. They are situated, in fact, at the domain-specific level, which is considered more appropriate for use in computational tasks (Gildea and Jurafsky 2002). However, given the too-restrictive constraints of the algorithm, the clustering method also overgenerates redundant clusters. In future work, we will attempt to reduce redundancy using clustering algorithms based on concept lattices (Kovacs and Baranyi 2002).

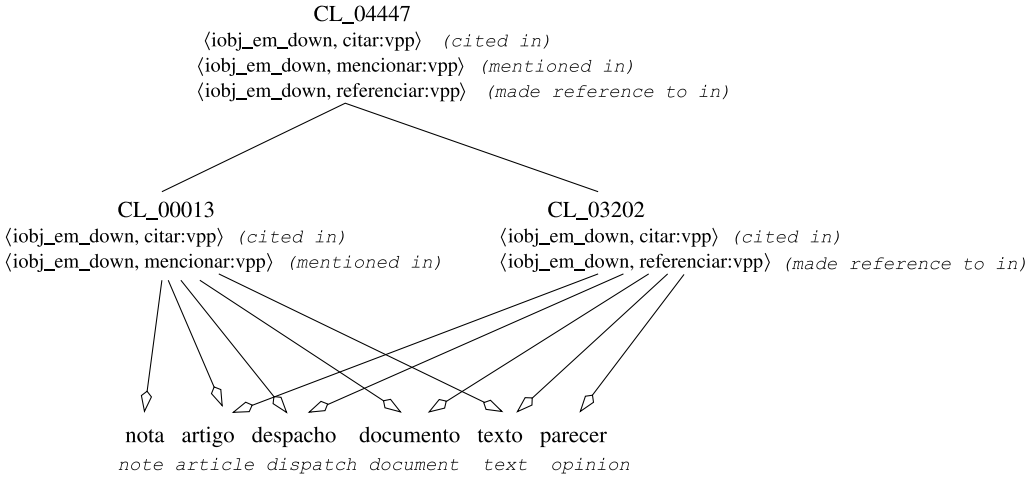

Figure 6
Clustering 2.

Table 6
Corpus data.

	Corpus PGR	Corpus EC
Word occurrences	6,643,579	3,110,397
Binary dependencies	966,689	487,916
Syntactic positions	178,522	113,847
Basic clusters	370,853	166,886
Clusters (Clustering 2)	16,274	10,537

In order to evaluate the linguistic relevance of these clusters, we check in section 8 whether they are useful in a parsing task. The degree of efficiency in such a task (parsing) may serve as a reliable evaluation for measuring the soundness of the learning strategy.

7.4 Related Clustering Methods

There are other approaches to acquisition of word senses by clustering words according to context-sensitive information. Similarly to our work, these approaches assume, on the one hand, that a word can appear in different clusters (soft clustering), and on the other hand, that each cluster represents a particular sense distinction of the words that are elements of it. Different clustering methods can be distinguished.

First, some methods compare the similarity between pairs of syntactic positions (and not pair of words) in order to generate clusters of syntactic positions, whose features are set of words (Allegrini, Montemagni, and Pirrelli 2003; Faure and Nédellec 1998; Reinberger and Daelemans 2003). Similarly to our approach, these methods follow both the relative view on word similarity and the assumption on contextual word sense, which were introduced above in sections 3.3 and 3.4, respectively. However, these methods differ from ours in several aspects. That of Reinberger and Daelemans (2003) does not use any kind of filtering process. So given a cluster of positions, the set of its features is basically defined as the union of their co-occurring words. This method turns out not to be appropriate when extracted co-occurrences are noisy. The cooperative system Asium presented in Faure and Nédellec (1998) filters out incorrect words from clusters of positions. However, unlike in our work, this task is not automatic. It requires manual removal of those words that have been incorrectly tagged or analyzed. Similarly to our approach, Allegrini, Montemagni, and Pirrelli (2000) developed an automatic procedure to remove odd words from clusters. It consists in defining a first clustering step in which positions are aggregated in basic clusters, which are called **substitutability islands**. As in Clustering 1 (section 7.1), each basic cluster selects only those words occurring in all positions of the cluster. However, Allegrini, Montemagni, and Pirrelli (2000) define a second clustering step involving significant differences with regard to our Clustering 2. Given a position p, they define a list of basic clusters containing p. This list is ranked and then used as the input to a clustering strategy that aggregates only basic clusters belonging to that list. So a cluster containing p cannot be aggregated with a cluster that does not contain p. This is a very strong constraint. It reduces significantly the system's ability to make generalizations.

Second, other methods discover word senses by clustering words according to the similarity of their whole distributions (Pantel and Lin 2002; Lin and Pantel 2001). These

methods, then, follow both the absolute view on word similarity and Harris's distributional hypothesis, which we introduced in section 2.3. However, in order to make the absolute view more relative, a collection of small and tight clusters (called **committees**) is proposed in a first step. These tight clusters are supposed to represent different word senses. Then in a second step, each word is assigned to a set of committees.

Finally, Pantel and Lin (2000) offer a hybrid method based on the two basic views on semantic similarity, absolute and relative. Given a word w occurring in position p, in any pair of type $\langle verb, function \rangle$ or $\langle noun, preposition \rangle$, the system, in a first step, generates classes of contextually similar words. A contextual class results from the intersection of the words occurring in p and the words similar to w. The definition of a contextual class contains the two views on word similarity. On the one hand, the words occurring in p are called the **cohorts** of w. The cohorts are similar to w only with regard to position p (relativized view). On the other hand, a corpus-based thesaurus is used to select words similar to w with regard to its whole position distribution (absolute view). Note that a contextual class is not far from what we call a basic cluster. In a second step, contextual classes are used to compute attachment association scores. The aim of the method is not to discover word senses (as in the methods outlined above), but to solve syntactic ambiguities. No clustering strategy is proposed to generate more general contextual senses.

Our system could also be considered a hybrid method, since besides the contextual hypothesis and the relative view, we also take into account the absolute view on word similarity to design a corpus-based thesaurus.

7.5 Automatic Thesaurus Construction

Clustering 2 uses a thesaurus of similar words to avoid undesirable aggregations. To design a corpus-based thesaurus, we follow the absolute view on word similarity: The similarity between two words is computed by comparing their whole context distribution. Our thesaurus is not specifically designed to be involved in the clustering process. It is designed primarily with the aim of measuring the discriminative capabilities of syntactic positions defined on the basis of corequirements (Gamallo et al. 2001). In particular, we check whether corequired positions are semantically more selective than those used by Grefenstette (1994), which were defined in terms of simple requirements. Experimental tests showed that corequirements permit a finer-grained characterization of "meaningful" syntactic positions.

To compute word similarity, we used the weighted version of the binary Jaccard measure defined in Grefenstette (1994). The weighted Jaccard similarity (WJ) between two words, w_1 and w_2, is computed by

$$WJ(w_1, w_2) = \frac{\sum_i \min(Assoc(w_1, p_i), Assoc(w_2, p_i))}{\sum_i \max(Assoc(w_1, p_i), Assoc(w_2, p_i))} \quad (20)$$

In (20), the weight $Assoc$ is the result of multiplying a local and a global weight, whose definitions are analogous to those given in formulas (17) and (18). The major difference is that in (20), positions are taken as attributes and words as objects.

We designed a particular thesaurus for each training corpus. As regards the PGR corpus, we produced 42,362 entries: 20,760 nouns, 16,272 verbs, and 15,330

adjectives. For each entry *w,* the thesaurus provides a list containing the 20 words most similar to *w*. This is the list that was later used in the clustering process.

8. Application and Evaluation

The acquired classes are used to solve attachment ambiguities. For this purpose, first, a lexicon is designed by using the linguistic information contained in the learned clusters. Then a particular heuristic uses this information to propose correct attachments. Some experiments are performed on two text corpora. The results are evaluated in section 8.3.

Table 7
Excerpt of entry secretário (*secretary*) (in the PGR corpus).

secretário (*secretary*)

SUBCAT
• ⟨*de_up, secretário*⟩ (*[_] of secretary*) =
cargo, carreira, categoria, competência, escalão, estatuto, funçao, remuneracão, trabalho, vencimento
(*post, career, category, qualification, rank, status, function, remuneration, job, salary*)
• ⟨*de_down, secretário*⟩ (*secretary of [_]*) =
administração, assembleia, autoridade, conselho, direcção, empresa, entidade, estado, governo, instituto, juiz, ministro, ministério, presidente, serviço, tribunal órgão
(*administration, assembly, authority, council direction, company, entity, state, government, institute, judge, minister, ministery, president, service, tribunal organ*)
• ⟨*iobj_a_up, secretário*⟩ (*[_] to the secretary*) =
aludir, aplicar:ref1, atender, atribuir, concernir, corresponder, determinar, presidir, recorrer, referir : ref1, respeitar
(*allude, apply, attend, assign, concern, correspond, determine, resort, refer, relate*)
• ⟨*iobj_a_up, secretário*⟩ (*[_] to the secretary*) =
caber, competir, conceder:vpp, conferir, confiar:vpp, dirigir:vpp, incumbir, pertencer
(*concern, be incumbent, be conceded, confer, be trusted, be sent, be incumbent, belong*)
• ⟨*iobj_por_up, secretário*⟩ (*[_] by the secretary*) =
assinar:vpp, conceder:vpp, conferir:vpp, homologar:vpp, louvar:vpp, subscrito
(*be signed, be conceded, be conferred, be homologated, be complimented, subscribe*)
• ⟨*lobj_up, secretário*⟩ (*the secretary [_]*) =
definir, estabelecer, fazer, fixar, indicar, prever, referir
(*define, establish, make, fix, indicate, foresee, refer*)

SENSE
• administração, assembleia, autoridade, chefe, comandante, comissão, conselho, director, direcção, entidade, estado, funcionário, gabinete, governador, governo, instituto, juiz, membro, ministro, ministério, presidente, provedor, secreteria, secretário, senhor, serviço, tribunal, órgão
(*administration, assembly, authority, chief, commander, commission, council, director, direction, entity, state, official, cabinet, governor, government, institute, judge, member, minister, ministry, president, purveyor, secretary, secretary, mister, service, tribunal, organ*)
• primeiro-ministro, autoridade, entidade, estado, membro, ministro, ministério, presidente, secretário
(*prime minister, authority, entity, state, member, minister, ministry, president, secretary*)

8.1 Design of a Lexicon with Corequirements

The learning method provides a lexicon with syntactic and semantic information. A word entry is divided into two types of information (see Table 7). SUBCAT is the repository of syntactic and semantic requirements. SENSE contains the different word sets to which the entry belongs. Each word set corresponds to a particular sense distinction. However, only the SUBCAT information is used here for the purpose of attachment resolution. Table 7 shows an excerpt of entry `secretário` (*secretary*). This entry is associated with a SUBCAT repository with six requirements and a SENSE repository containing two word senses.

The word `secretário` requires two nominal and four verbal arguments. Concerning the nominal positions, we learn that *secretary* selects for nouns such as *post* or *rank* in the *de_up* location, whereas it requires a class of nouns denoting institutions or functions in the *de_down* location. Concerning the verbal positions, we also learn that *secretary* requires various verb classes in different verbal positions: two classes in location *iobj_a_up*, one class in *iobj_por_up*, and one more in *lobj_up*.

A syntactic pattern of subcategorization arguments underlies the organization of the SUBCAT repository in Table 7. This pattern can be represented as follows:

$$(X_v\, a_{prep}\, \alpha_n)_{vp} \;\vee\; (Y_v\, por_{prep}\, \alpha_n)_{vp} \;\vee\; (Z_n\, de_{prep}\, \alpha_n)_{np} \;\vee\; (\alpha_n\, de_{prep}\, W_n)_{np} \;\vee\; (\alpha_n\, U_v)_{vp} \quad (21)$$

where X, Y, Z, ... stand for variables of subcategorized words, while α is the subcategorizer. If α is `secretário`, then the specific values of X, Y, Z, ... can be found in Table 7. For example, according to Table 7, the noun `cargo` instantiates Z, while the verb `pertencer` instantiates X. The symbol $\vee$ stands for Boolean disjunction. We take into consideration that at least in Portuguese, all word arguments are optional. Even the subject of a verb may be omitted. Note, however, that the syntactic pattern in (21) does not allow it to be distinguished whether arguments are compatible or not. For instance, it is not able to predict that $(Y_v\ por_{prep}\ \alpha_n)_{vp}$ and $(\alpha_n\ U_v)_{vp}$ are argument positions that cannot appear in the same sentence. Moreover, there are no restrictions on the linear order of arguments. As we do not learn this type of syntactic information, the pattern depicted in (21) can be viewed merely as a set of potential arguments of a word. So our method does not allow a set of entirely organized subcategorization frames to be captured for each word.

Note that it is the corequirement structure that allows us to acquire a great number of requirement positions that are not usual in most standard approaches. Five positions of *secretary* require not standard dependent complements, but different types of heads. This is a significant novelty of our approach. Consider the positions that impose nonstandard requirements (i.e., nonstandard predicates). According to the standard definition of predicate given in section 4.2.1 (simple requirement definition), only locations *robj_down, lobj_down,* and *mod_up* give rise to positions with requirements.[8] By contrast, positions defined by the complementary locations (*robj_up, lobj_up, mod_down*) are considered mere complements of verbs or objects modified by adjectives. So they cannot impose any requirement, and thereby they are not semantically defined as predicates. In opposition to this viewpoint, our system learns more classes of requirements imposed by positions considered nonstandard predicates (5,192) than classes of requirements imposed by positions considered standard

8 Positions with prepositions are not taken into account in this analysis because they are ambiguous.

predicates (4,600). These experimental results seem to prove that nonstandard predicates correspond to positions with requirements. In sum, we may infer that binary dependencies are structured by corequirements.

Consider now the SENSE repository in Table 7. It contains two word sets which should represent two senses of `secretário`. Unfortunately, our clustering algorithm generates some redundancy. In this case, the two clusters should have been merged into one, since they seem to refer to the same concept. Cluster redundancy is the major problem of our learning strategy.

8.2 Attachment Heuristic CR

The syntactic and semantic requirements provided by the lexical entries are used to improve a parser and the DCG grammar it is based on. The description of the parser remains beyond the scope of this article; it has been described in Rocio, de la Clergerie, and Lopes (2001). Details of a symbolic DCG grammar with information on linguistic corequirements can be found in Gamallo, Agustini, and Lopes (2003). In this article, we only outline how the grammar uses this information to resolve syntactic attachments. Corequirements are at the center of attachment resolution. They are used to characterize a particular heuristic on syntactic attachment. This heuristic referred to as CR, is supposed to be more precise than RA. It states that two chunks are syntactically and semantically attached only if one of the following two conditions is verified: Either the dependent is semantically required by the head or the head is semantically required by the dependent. Take the expression

...`compete a o secretário`... (*is incumbent on the secretary*) (22)

This expression is analyzed as a *vp–pp* construction only if at least one of the two following requirements is satisfied:

Down requirement: The context ⟨*iobj_a_down, competir*⟩ (*be-incumbent on [_]*) requires a class of nouns to which `secretário` (*secretary*) belongs.

Up requirement: The context ⟨*iobj_a_up, secretário*⟩ (*[_] on secretary*) requires a class of verbs to which `competir` (*be incumbent*) belongs.

Corequirements are viewed here as constraints on the syntactic rules of a symbolic grammar. Attachments are then solved by using Boolean, and not purely probabilistic, constraints. According to the lexical information illustrated in Table 7, expression (22) can be analyzed as a *vp–pp* construction because at least the *up* requirement is satisfied. Note that even if we had no information on the verb requirements, the attachment would be allowed, since the noun requirements in the dependent (*up*) location were learned. So we learned that the noun `secretário` has as argument the verb `competir` in location ⟨*iobj_a_up*⟩. As we show in the evaluation procedure, corequirements are also used to resolve long-distance attachments.

8.3 Evaluating Performance of Attachment Resolution

We evaluated the performance of CR, that is, the attachment heuristic based on Boolean corequirements. The general aim of this evaluation was to check whether the

linguistic requirements we learned were adequate for use in a parsing task. The degree of efficiency in such a task may serve as a reliable evaluation for measuring the soundness of our learning strategy.

8.3.1 Test Data. The test data consisted of sequences of basic phrases (i.e., chunks). The phrase sequences selected belong to three types: *vp–np–pp*, *vp–pp–pp*, and *np–pp–pp*. They were randomly selected from two different (and already chunked) test corpora: a group of 633 sequences was selected from the EC corpus and another group of 633 was selected from PGR. Each group of 633 sequences was constrained to have three equal partitions: 211 *vp–np–pp* sequences, 211 *vp–pp–pp* sequences, and 211 *np–pp–pp* sequences. The test corpus from which each group was selected had previously been separated from the training corpus, so the sequences used for the test were excluded from the learning process. Then the annotators (the coauthors) manually proposed the correct attachments for each phrase sequence, using the full linguistic context. Some specific instructions were given to the annotators for the most controversial cases. The following excerpts from these instructions are illustrative: (1) if a *pp* seems to be a modifier of the verb, then it is attached to the *vp*; (2) if a *pp* is a modifier of the sentence, no attachment is proposed; (3) if an *np* following a *vp* is either the direct object or the subject of the verb, then the *np* is attached to the *vp*; (4) if a *pp* seems to be attached to two phrases, two attachments are proposed (we retain the ambiguity); (5) if a phrase contains a word that was not correctly tagged, no attachment is proposed. Note that verbal modifiers and verbal complements are treated in the same way (see subsection 4.2.2). Moreover, we consider a *robj* (i.e., an *np* following a *vp*) as being able to be instantiated by two different functions: both a direct object and a subject (section 6.1).

Most works on attachment resolution use as test data only phrase sequences of type *vp–np–pp* (Sekine et al. 1992; Hindle and Rooth 1993; Ratnaparkhi, Reymar, and Roukos 1994; Collins and Brooks 1995; Li and Abe 1998; Niemann 1998; Grishman and Sterling 1994). These approaches consider each sequence selected for evaluation as having the potential to be syntactically ambiguous in two ways. For instance, the seuqence of chunks

$$[_{VP}\, cut][_{NP}\, the\ potato][_{PP}\, with\ a\ knife] \tag{23}$$

can be elaborated either by the parse

$$[_{VP}\, cut[_{NP}\, the\ potato[_{PP}\, with\ a\ knife]]] \tag{24}$$

which represents a syntactic configuration based on proximity (*phrase2* is attached to *phrase1* and *phrase3* is attached to *phrase2*), or by

$$[_{VP}\, cut[_{NP}\, the\ potato][_{PP}\, with\ a\ knife]] \tag{25}$$

which is here the correct configuration. It contains both a contiguous and a long-distance attachment: *phrase2* is attached to *phrase1* and *phrase3* is attached to *phrase1*.

Table 8
Different types of syntactic sequences and various types of syntactic ambiguities.

np–pp–pp	[$_{np}$ o artigo relativo] [$_{pp}$ a o decreto] [$_{pp}$ de a lei] (*the article referring to the decree-law*)
vp–pp–pp	[$_{vp}$ publicou] [$_{pp}$ em os estatutos anexos] [$_{pp}$ a o citado diploma] (*published in the statutes appended to the referred diploma*)
vp–np–pp	[$_{vp}$ tem] [$_{np}$ acesso] [$_{pp}$ em a medida] (*has access insofar as*)

We consider, however, that the process of attachment resolution can be generalized to other syntactic sequences and ambiguity configurations. On the one hand, we evaluated not only one, but three types of phrase sequences: *vp–np–pp*, *vp–pp–pp*, and *pp–pp–pp*. On the other hand, these sequences cannot be reduced to only two syntactic configurations (two parses). They can be syntactically ambiguous in different ways. These ambiguities are introduced by adjective arguments and sentence adjuncts (see Table 8).

Table 8 shows phrase sequences that cannot be analyzed by means of the two standard configurations underlying parses (24) and (25). None of the sequences in that table match the two standard configurations. For instance, a o decreto (*to the decree*), which is the *phrase2* of the first example, is not attached to the head of *phrase1*, but to the adjective relativo (*referring*). Similarly, in the second expression, a o citado diploma (*to the referred diploma*) is attached to the adjective anexos (*appended*) and not to the head of *phrase2*. Subcategorization of adjectives introduces a new type of structural ambiguity, which makes it more difficult to make attachment decisions. Finally, in the third sequence, em a medida (*insofar as*) is the beginning of an adverbial sentence, so it is not attached to one of the individual phrases but to the whole previous sentence. In sum, resolving structural ambiguity cannot be reduced to a binary choice between the two configurations depicted in (24) and (25). We return to this matter below.

Another important property of test data is that they contain incorrectly tagged words. We do not remove these cases, since they can give us significant information about how (in)dependent of noisy data our learning method is.

8.3.2 The Evaluation of Protocol. Each sequence selected from the test corpus contains three phrases and two candidate attachments. So given a test expression, two different attachment decisions are evaluated:

Decision A: Is *phrase2* attached to *phrase1,* or not attached at all?

Decision B: Is *phrase3* attached to *phrase2,* attached to *phrase1,* or *not attached at all*?

As we selected 633 $\times$ 2 test expressions, and each expression implicitly contains two attachment decisions, the total number of decisions that we evaluated was 2,532. By contrast, in most related approaches, test expressions are ambiguous in only two senses: *phrase3* is attached to either *phrase2* or *phrase1*. Such approaches do not consider

the attachment between *phrase2* and *phrase1*. So in these approaches, Decision A is not taken into account. Moreover, they do not evaluate those cases in which *phrase3* is not attached to *phrase2* or to *phrase1*. In sum, only one decision per expression is evaluated, namely, the decision concerning the PP-attachment. This type of evaluation, however, is not appropriate to measure the capability of the system to identify the nonstandard structural ambiguities described above (section 8.3.1). For instance, we expect the system not to propose that the *pp* `ao diploma` *(to the [referred] diploma)* is attached to the previous *np*, headed by `estatutos`, in the second example of Table 8. The correct decision is to propose no attachment between the *pp* (*phrase3*) and either of the two previous phrases taking part in the sequence *vp–pp–pp*. The attachment is actually to a word, namely, the adjective `anexo`, which is not a direct constituent of the abstract sequence *vp–pp–pp*.

Another important aspect of the evaluation protocol is that CR overgenerates attachments. There are several cases in which the three phrases of a sequence are semantically related. In those cases, CR often proposes three attachments even if only two of them are syntactically allowed. For instance, take the following *np–pp–pp* sequence:

$$[_{np}\texttt{a remuneração}]\ [_{pp}\texttt{de o cargo}]\ [_{pp}\texttt{de secretário}] \quad (26)$$
(*the salary concerning the post of secretary*)

which would be correctly analyzed by using the same configuration as in parse (24), that is,

$$[_{np}\texttt{a remuneração}\ [_{pp}\texttt{de o cargo}\ [_{pp}\texttt{de secretário}]]] \quad (27)$$

Note that there exists a strong semantic relationship between *phrase3* (`de secretário`) and *phrase1* (`a remuneração`), even if they are not syntactically attached in (27). Taking into account the semantic requirements stocked in the lexicon (see Table 7), CR is induced to propose, in addition to the two correct attachments, a long-distance dependency, which seems not to be syntactically correct in this particular case. We call this phenomenon **attachment overgeneration**. When a sequence contains two semantically related phrases that are not actually syntactically dependent, CR overgenerates an additional attachment. Attachment overgeneration was found in ≈15% of expressions selected from the test corpus. In order to overcome this problem, we use a default rule based on right association. The default rule removes the long-distance attachment and proposes only the two contiguous ones. This simple rule has an accuracy of more than 90% with regard to the 15% of sequences containing overgeneration.

From a semantic viewpoint, attachment overgeneration seems not to be a real problem. The semantic interpretation of sequence (26) needs to account for all conceptual relations underlying the sequence. So the semantic requirements that linked `secretário` to `remuneração` (even if they are not syntactically dependent) are useful for building a semantic representation of the sequence.

8.3.3 Baseline (RA). Concerning the ability to propose correct syntactic attachments, we made a comparison between CR and a baseline strategy. As a baseline, we used the

attachments proposed by right association. For each sequence of the test data, RA always proposes the configuration underlying parses (27) and (24); that is, *phrase2* is attached to *phrase1*, and *phrase3* is attached to *phrase2*.

8.3.4 Similarity-Based Lexical Association. We also compared CR to a very different learning strategy: the similarity-based lexical method (Sekine et al. 1992; Grishman and Sterling 1994; Dagan, Marcus, and Markovitch 1995; Dagan, Lee, and Pereira 1998), described in section 2.3. We simulated here a particular version of this strategy. First, we used the log-likelihood ratio as a score of the association between pairs of syntactic positions and words. We restricted the lexical association procedure to suggest attachments only in cases in which the absolute value of the ratio was greater than an empirically set threshold (−3.00). Then, in order to generalize from unobserved pairs, a list of similar words was used to compute nonzero association scores. For this purpose, the thesaurus described in section 7.5 turned out to be useful.

According to Dagan, Marcus, and Markovitch (1995), the similarity-based lexical association LA_{sim} between position p and word w is obtained by computing the average of the likelihood ratios between p and the k most similar words to w:

$$LA_{sim}(p, w) = \frac{\sum_{i=0}^{k} LA(p, w_i)}{NZ} \tag{28}$$

where $LA(p, w_i)$ is the likelihood ratio between p and one of the k most similar words to w. NZ represents the number of nonzero values among $LA(p, w_1)$, $LA(p, w_2)$,..., $LA(p, w_k)$.

Corequirements are also considered. Given dependency (*robj*; `ratificar`$^{\downarrow}$, `lei`$^{\uparrow}$) (*ratify the law*), we compute the two following lexical associations:

$$\begin{array}{l} LA_{sim}(\langle robj\_down,\ ratificar\rangle, \texttt{lei}) \\ LA_{sim}(\langle robj\_up, lei\rangle \texttt{ratificar}) \end{array} \tag{29}$$

The scores of these two associations are taken into account in the evaluation procedure. In particular, the sum of the two scores (if each of them is greater than the empirically set threshold) will be used to make a decision on the attachment between an *np* headed by `lei` and a *vp* headed by `ratificar`.

8.3.5 Precision and Recall. The evaluation of each attachment decision made by the system can be

- True positive (*tp*): The system proposes a correct attachment;
- True negative (*tn*): The system proposes correctly that there is no attachment;
- False positive (*fp*): The system proposes an incorrect attachment;
- False negative (*fn*): The system proposes incorrectly that there is no attachment.

We refer to both *tp* and *tn* as **true decisions** (*td*). The evaluation test measures the ability of the system to make true decisions. As far as our strategy and the similarity-

based approach are concerned, a false negative (*fn*) is interpreted as the situation in which the system lacks sufficient subcategorization information to make a decision. By contrast, the baseline always proposes an attachment.

Taking into account these variables, **precision** is defined as the number of true decisions suggested by the system divided by the number of total suggestions. That is:

$$\text{precision} = \frac{td}{td \ + \ fp} \tag{30}$$

Recall is computed as the number of true decisions suggested by the system divided by all the decisions that have been made (i.e., the total number of ambiguities):

$$\text{recall} = \frac{td}{td \ + \ fp \ + \ fn} \tag{31}$$

To clarify the evaluation procedure, Table 9 displays the different attachment decisions made on the following test sequence:

$$[_{vp}\ \texttt{assistir}\ [_{pp}\ \texttt{por o representante}\ [_{pp}\ \texttt{de o Estado}-\texttt{Membro}]]] \\ (\textit{assisted by the delegate of the Member-State}) \tag{32}$$

The two correct attachments in (32), proposed by the human annotator, are compared against the attachment decisions proposed by the three methods at stake: heuristic with Boolean corequirements, similarity-based lexical association, and right association, which is the baseline. Table 9 assesses the two different decisions (A and B) made by each method. Note that both CR and LA_{sim} take advantage of corequirements. Indeed, each decision is made after two types of subcategorization information have been considered: the requirements the dependent word must satisfy and the requirements that the headword must satisfy.

Decision A concerns the first candidate attachment, that is, the dependency between [$_{vp}$ `assistir`] and [$_{pp}$ `por o representante`]. Let us analyze the behavior of the three methods. LA_{sim} incorrectly suggests that there is no attachment. The score of two internal requirements is zero, so the final decision is a false negative: *fn*. The system has no information on requirements because on the one hand, the two phrases at stake do not co-occur in the training corpus, and on the other, co-occurrences of phrases with similar words were not attested (and then no generalization was allowed). CR, by contrast, is endowed with the appropriate requirements to correctly suggest an attachment (*tp*) between the two phrases, even though they are not attested in the training corpus. The clustering strategy allowed the system to learn both that ⟨*iobj_por_D*, *assistir*⟩ requires `representante` and that ⟨*iobj_por_H*, *representante*⟩ requires `assistir`. Note that in order to suggest the attachment, it is not necessary that the two complementary requirements be learned. As has been noted in section 8.2, the learning of only one of them is enough to make the suggestion. Finally, RA also suggests the correct attachment. Indeed, the two phrases in (32) are related by right association.

Table 9
Evaluation of a test sequence.

CR	Decision A: ⟨*iobj_por_D*, *assistir*⟩ requires `representante`: YES ⟨*iobj_por_H*, *represantante*⟩ requires `assistir`: YES Result: *tp* Decision B: ⟨*iobj_por_D*, *assistir*⟩ requires `Estado-Membro`: NO ⟨*iobj_por_H*, *Estado–Membro*⟩ requires `assistir`: NO ⟨*de_D*, *representante*⟩ requires `Estado-Membro`: YES ⟨*de_H*, *Estado – Membro*⟩ requires `representante`: YES Result: *tp*
LA_{sim}	Decision A: LA_{sim} (⟨*iobj_por_D*, *assistir*⟩, `representante`): 0 LA_{sim} (⟨*iobj_por_H*, *representante*⟩, `assistir`): 0 Result: *np* Decision B: LA_{sim} (⟨*iobj_por_D*, *assistir*⟩, `Estado-Membro`): 0 LA_{sim} (⟨*iobj_por_H*, *Estado – Membro*⟩, `assistir`): 0 LA_{sim} (⟨*de_D*, *representante*⟩, `Estado-Membro`): 136.70 LA_{sim} (⟨*de_H*, *Estado – Membro*⟩, `representante`): 176.38 Result: *tp*
RA	Decision A: [$_{vp}$ assistir [$_{pp}$ por o representate]]: YES Result: *tp* Decision B: [$_{pp}$ por o representate [$_{pp}$ de os Estados-Membros]]: YES Result: *tp*

As regards Decision B, all three methods correctly suggest that there is an attachment (*tp*) between [$_{np}$ `o representate`] and [$_{pp}$ `de o Estado-Membro`].

8.3.6 Results. Table 10 reports the test scores in regard to the precision and recall from the experiments performed. These scores concern three methods, namely RA, LA_{sim}, and CR, two text corpora (EC and PGR), and three types of phrase sequences. There are no significant differences between the scores obtained from corpus EC and those from PGR, CR, for instance, obtains very similar *F*-scores over the two corpora. However, there are important differences among the precision values associated with the three phrase sequences. In particular, the scores on sequence *vp–np–pp* are significatively higher than those on the other sequences, regardless of the method employed. This is motivated by the fact that in most *vp–np–pp* sequences (≈95%), there is a true attachment between *np* and *vp*. So the precision score achieved by the three methods with regard to this particular attachment decision is very high. Prepositional-phrase attachments, by contrast, are more ambiguous, which causes sequences *vp–pp–pp* and *np–pp–pp* to be less predictable. Indeed, such sequences have two prepositional phrases involved in the attachment decisions.

Table 10
Evaluation taking into account three types of sequences and two corpora: *EC* and *PGR*. Pr = precision, Rec = recall, and F-S = *F*-score.

	Baseline (RA)					
Sequences	Pr_{EC}	Pr_{PGR}	Rec_{EC}	Rec_{PGR}	$F\text{-}S_{RV}$	$F\text{-}S_{PGR}$
np–pp–pp	.71	.72	.71	.72	.71	.72
vp–np–pp	.83	.80	.83	.80	.83	.80
vp–pp–pp	.75	.74	.75	.74	.75	.74
	Lexical association (LA_{sim})					
Sequences	Pr_{EC}	Pr_{PGR}	Rec_{EC}	Rec_{PGR}	$F\text{-}S_{RV}$	$F\text{-}S_{PGR}$
np–pp–pp	.77	.82	.66	.72	.71	.77
vp–np–pp	.90	.86	.75	.74	.82	.79
vp–pp–pp	.85	.89	.65	.70	.74	.78
	Boolean requirements (CR)					
Sequences	Pr_{EC}	Pr_{PGR}	Rec_{EC}	Rec_{PGR}	$F\text{-}S_{RV}$	$F\text{-}S_{PGR}$
np–pp–pp	.85	.86	.73	.76	.78	.81
vp–np–pp	.92	.93	.75	.78	.83	.85
vp–pp–pp	.86	.91	.69	.75	.77	.82

Concerning the differences among the three methods, Table 11 averages the results of the three methods over the two corpora and the three phrase sequences. The total precision of our method (CR) reaches 0.89, that is, four percentage points higher than LA_{sim}. Note that the precision value of LA_{sim} is not far from the values reached by other approaches to attachment resolution based on the similarity-based lexical association strategy. For instance, the method described in Grishman and Sterling (1994) scores a precision of ≈0.84. Concerning recall, CR also reaches a level of precision that is four points higher than that achieved by LA_{sim}. This entails that on the one hand, the ability of CR to learn accurate subcategorization information is higher than that of LA_{sim}, and on the other hand, the ability of CR to learn from sparse data and to generalize is at least no lower than that of LA_{sim}.

The baseline score informs us that about 76% of attachments are links by proximity. The remainder (24%) are either long-distance attachments between *phrase3* and *phrase1*, other types of attachments such as adjective complements, and sentence modifiers, or finally, tagger errors. Note that there is no difference between the baseline method's precision and recall scores. Since RA always makes a (true or false) positive decision, there cannot be (true or false) negative decisions.

Table 11
Total scores of the three methods. For each method, we compute the average of the three sequences and the two corpora.

	Precision	Recall	*F*-score
Baseline	.76	.76	.76
LA_{sim}	.85	.71	.77
CR	.89	.75	.81

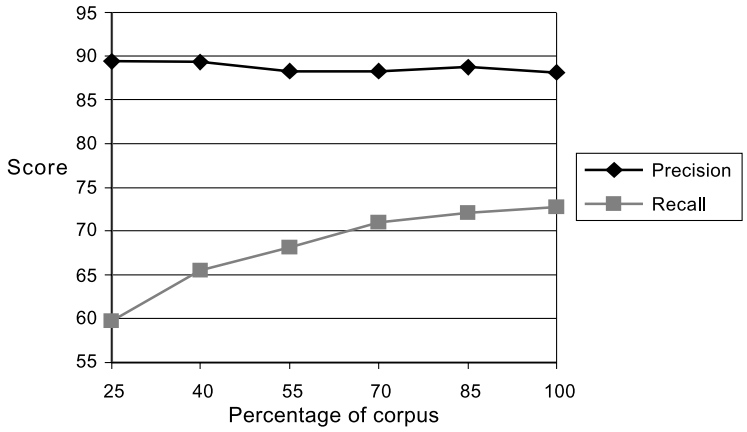

Figure 7
Variation of recall and precision as a function of corpus size.

Some tagger errors, especially those that appear systematically and regularly in the training corpus, have a negative influence on the precision of both LA_{sim} and CR. These methods are sensitive to noisy data.

In order to measure recall and precision stability, we ran the clustering process over six partitions (25%, 40%, 55%, 70%, 85%, and 100%) of the EC corpus. Figure 7 shows that recall improves with corpus size. However, recall growth is more significant in smaller partitions. In this particular corpus, recall stability seems to be achieved when the corpus contains three million words. It follows that in order to improve recall, we would have to use not only a bigger training corpus, but also a more efficient clustering strategy, that is, a strategy that would be able to make additional correct generalizations. Finally, note that precision neither increases nor decreases with corpus size.

9. Conclusion and Future Work

This article has presented a particular unsupervised strategy to automatically acquire syntactic and semantic requirements. Our aim was to learn two types of information about a given word: the syntactic positions in which the word appears and the semantic requirements associated with each syntactic position. Besides that, this strategy also allowed us to discriminate word senses. The strategy is based on several linguistic assumptions. First, it was assumed that not only does the syntactic head impose restrictions on its dependent word, but also that the dependent word may select for a specific type of head, a phenomenon referred to as corequirement. Second, we claimed that similar syntactic positions share the same semantic requirements. So we measured not similarity between words on the basis of their syntactic distribution, but similarity between syntactic positions on the basis of their word distribution. It was assumed that the latter kind of similarity conveys more pertinent information on linguistic requirements than the former one. The learning process allowed us to provide a lexicon with, among other information, both syntactic subcategorization and selection restrictions. This information was used to constrain attachment heuristics.

In current work, we are using the learned clusters in other NLP applications than attachment resolution. They are being used to automatically select word senses in context (word sense disambiguation task). For this purpose, we are performing new

experiments on less domain-specific text corpora, since such corpora increase the number of senses per word. On the other hand, these clusters turn out to be very useful for checking whether two or more different morphological forms of a word are semantically related. For instance, if *ratification of [_]* is similar to *ratify [_]*, we may infer that the verb *ratify* and the noun *ratification* are semantically related.

In future work, we aim at extending the lexicon in order to increase the coverage of the parser. To do this, parsing and learning can be involved in a bootstrapping process. The dependencies proposed by heuristic CR will be used as input to discover new linguistic requirements. This new information will enable us to update the lexicon, and then to propose new dependencies. At each cycle, the lexicon will be provided with new requirements, and thereby the parser coverage will be higher. The successive "learning + parsing" cycles will stop as no more new information is acquired and no more new dependencies are proposed.

Acknowledgments

The work by Pablo Gamallo was supported by a grant from Portugal's Fundação para a Ciência e Tecnologia (ref: PRAXIS XXI/ BPD/2209/99). Alexandre Agustini was supported by Brazil's Federal Agency for Post-Graduate Education (CAPES) and Pontifical Catholic University of Rio Grande do Sul (PUCRS).

References

Allegrini, Paolo, Simonetta Montemagni, and Vito Pirrelli. 2000. Learning word clusters from data types. In *COLING-00*, pages 8–14, Saarbrücken, Germany.

Allegrini, Paolo, Simonetta Montemagni, and Vito Pirrelli. 2003. Example-based automatic induction of semantic classes through entropic scores. In A. Zampolli, N. Calzolari, and L. Cignoni, editors, *Computational Linguistics in Piza—Linguistica Computazionale a Pisa* (special issue of *Linguistica Computazionale*). Pisa and Rome: IEPI, volume 1, pages 1–45.

Androutsopoulos, Ion and Robert Dale. 2000. Selectional restrictions in HPSG. In *Eighteenth Conference on Computational Linguistics (COLING)*, pages 15–20, Saarbrücken, Germany.

Basili, Roberto, Maria Pazienza, and Paola Velardi. 1992. Computational lexicons: The neat examples and the odd exemplars. In *Proceedings of the Third ANLP*, Trento, Italy.

Beale, Stephen, Sergei Niremburg, and Evelyn Viegas. 1998. Constraints in computational semantics. In *COLING-98*, Montreal.

Brent, Michel R. 1991. Automatic acquisition of subcategorization frames from untagged text. In *29th Annual Meeting of ACL*, pages 209–214, Berkeley, CA.

Ciaramita, Massimiliano and Mark Johnson. 2000. Explaining away ambiguity: Learning verb selectional preference with Bayesian networks. In *COLING-00*, pages 187–193, Saarbrücken, Germany.

Collins, Michael and James Brooks. 1995. Prepositional phrase attachment through a backed-off model. In *Proceedings of the Third Workshop on Very Large Corpora*, pages 27–38, Cambridge, MA.

Dagan, Ido, Lillian Lee, and Fernando Pereira. 1998. Similarity-based methods of word cooccurrence probabilities. *Machine Learning*, 43:56–63.

Dagan, Ido, Shaul Marcus, and Shaul Markovitch. 1995. Contextual word similarity and estimation from sparse data. *Computer Speech and Language*, 9(2):123–152.

de la Clergerie, Eric. 2002. Construire des analyseurs avec dyalog. In *Proceedings of TALN'02*, Nancy, France.

Faure, David and Claire Nédellec. 1998. Asium: Learning subcategorization frames and restrictions of selection. In *ECML98, Workshop on Text Mining*, Chemnitz, Germany.

Framis, Francesc Ribas. 1995. On learning more appropriate selectional restrictions. In *Proceedings of the Seventh Conference of the European Chapter of the Association for Computational Linguistics*, Dublin.

Gamallo, Pablo. 2003. Cognitive characterisation of basic grammatical structures. *Pragmatics and Cognition*, 11(2):209–240.

Gamallo, Pablo, Alexandre Agustini, and Gabriel P. Lopes. 2003. Learning subcategorisation information to model a grammar with co-restrictions. *Traitement Automatic de la Langue*, 44(1):93–117.

Gamallo, Pablo, Caroline Gasperin, Alexandre Agustini, and Gabriel P. Lopes. 2001. Syntactic-based methods for measuring word similarity. In V. Mautner, R. Moucek, and K. Moucek, editors, *Text, Speech, and Discourse (TSD-2001)*. Berlin: Springer Verlag, pages 116–125.

Gildea, Daniel and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Grefenstette, Gregory. 1994. *Explorations in Automatic Thesaurus Discovery*. Kluwer Academic, Norwell, MA.

Grishman, Ralph and John Sterling. 1994. Generalizing automatically generated selectional patterns. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, Kyoto, Japan.

Harris, Zellig. 1985. Distributional structure. In J. J. Katz, editor, *The Philosophy of Linguistics*. New York: Oxford University Press, pages 26–47.

Hindle, Donald and Mats Rooth. 1993. Structural ambiguity and lexical relations. *Computational Linguistics*, 19(1):103–120.

Hudson, Richard. 2003. The psychological reality syntactic dependency relations. In *MTT2003*, Paris.

Kovacs, Laszlo and Peter Baranyi. 2002. Document clustering based on concept lattice. In *IEEE International Conference on System, Man and Cybernetics (SMC'02)*, Hammamet, Tunisia.

Langacker, Ronald W. 1991. *Foundations of Cognitive Grammar: Descriptive Applications*, volume 2. Stanford University Press, Stanford.

Li, Hang and Naoki Abe. 1998. Word clustering and disambiguation based on co-occurrence data. In *COLING-ACL'98*, pages 749–755, Montreal.

Lin, Dekang. 1998. Automatic retrieval and clustering of similar words. In *COLING-ACL'98*, pages 768–774, Montreal.

Lin, Dekang and Patrick Pantel. 2001. Induction of semantic classes from natural language text. In *SIGKDD-01*, San Francisco.

Manning, Christopher. 1993. Automatic acquisition of a large subcategorization dictionary from corpora. In *31st Annual Meeting of ACL*, pages 235–242, Columbus, OH.

Manning, Christopher and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA.

Marques, Nuno and Gabriel P. Lopes. 2001. Tagging with small training corpora. In F. Hoffmann, D. Hand, N. Adams, D. Fisher, and G. Guimaraes, editors, *Advances in Intelligent Data Analysis*. Lecture Notes in Computer Science. Springer Verlag, Berlin, pages 62–72.

Marques, Nuno, Gabriel P. Lopes, and Carlos Coelho. 2000. Mining subcategorization information by using multiple feature loglinear models. In *10th CLIN*, pages 117–126, UILOTS Utrecht.

Niemann, Michael. 1998. Determining PP attachment through semantic associations and preferences. In *ANLP Post Graduate Workshop*, pages 25–32, Sydney.

Pantel, Patrick and Dekan Lin. 2000. An unsupervised approach to prepositional phrase attachment using contextually similar words. In *ACL'00*, pages 101–108, Hong Kong.

Pantel, Patrick and Dekan Lin. 2002. Discovering word senses from text. In *Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 613–619, Edmonton, Alberta, Canada.

Pustejovsky, James. 1995. *The Generative Lexicon*. MIT Press, Cambridge, MA.

Ratnaparkhi, Adwait, Jeff Reymar, and Salim Roukos. 1994. A maximum entropy model for prepositional phrase attachment. In *Proceedings of the ARPA Human Language Technology Workshop*, pages 250–255, Princeton, NJ.

Reinberger, Marie-Laure and Walter Daelemans. 2003. Is shallow parsing useful for unsupervised learning of semantic clusters? In *Fourth Conference on Intelligent Text Processing and Computational Linguistics (CICLing-03)*, pages 304–312, Mexico City.

Resnik, Philip. 1997. Selectional preference and sense disambiguation. In *ACL-SIGLEX Workshop on Tagging with Lexical Semantics*, Washington, DC.

Rocio, Vitor, Eric de la Clergerie, and Gabriel Lopes. 2001. Tabulation for multi-purpose partial parsing. *Journal of Grammars*, 4(1):41–65.

Schütze, Hinrich. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–124.

Sekine, Satoshi, Jeremy Carrol, Sofia Ananiadou, and Jun'ichi Tsujii. 1992. Automatic learning for semantic collocation. In *Proceedings of the Third Conference on Applied Natural Language Processing*, pages 104–110, Trento, Italy.