Comparing Window and Syntax Based Strategies for Semantic Extraction[⋆]

Pablo Gamallo Otero

Departamento de Língua Espanhola, Faculdade de Filologia
Universidade de Santiago de Compostela, Galiza, Spain
pablogam@usc.es

Abstract. In this paper, we describe and compare two different approaches for extracting similar words from large corpora. In particular, we compared a method based on syntactic contexts with two strategies relying on windows of tagged words, one using word order and the other bags of words. On a Portuguese corpus of 12 million words, syntactic contexts produce significantly better results for both frequent and not very frequent words.

1 Introduction

Finding semantically related words from large text corpora is one of the most popular tasks in Information Extraction. This is required to achieve more ambitious objectives, such as thesaurus construction, ontology design, question-answering enrichment, etc. The basic idea underlying the different techniques to find semantic similarity states that words are semantically related if they share a large number of contexts. There are basically two methods for defining word contexts. On the one hand, the context of a word is defined as the n words surrounding it (n-grams), where n stands for a window size. The methods using this type of word contexts are known as window-based approaches. On the other hand, the context of a word is determined by grammatical dependency relations. In this case, contexts are defined making use of syntax-based techniques.

It is broadly assumed that window-based approaches offer some advantages with regard to syntactic strategies: concerning *speed*, they are much less time consuming, while parsing large corpora is expected to be less computationally efficient. As far as *portability* is concerned, windowing techniques do not require contexts to be defined using specific grammars of particular natural languages; they are not language dependent. In addition, it has not been clearly demonstrated that syntactic contexts perform better that window contexts for discovering word similarity. On the contrary, it is assumed that the semantic relationships generated by approaches which use windowing techniques put words

⋆ This work has been supported by Ministerio de Educació y Ciencia of Spain, within the project ExtraLex, ref: PGIDIT07PXIB204015PR.

together according to associative relations, e.g., *doctor* and *hospital*. These relations are difficult to grasp by syntactic based methods, since related words such as *doctor* and *hospital* do not appear in the same syntactic contexts.

In this paper, we propose a syntax-based method which is provided with some of the advantages of windowing techniques: it is computationally efficient since the parsing strategy is robust and uses basic regular expressions. It is not language dependent since it relies on a multilingual parser whose grammar consists of very generic rules aimed to analyze texts in several languages. On the other hand, unlike window techniques, it is not aimed at discovering generic semantic associations between words, but only relationships between words belonging to the same class/kind of entities (i.e., *co-hyponyms*). We will demonstrate that methods using syntactic information have the tendency to find similarities between words that belong to the same semantic class, e.g., *doctor* and *nurse*, *teacher* and *pupil*. This specific semantic information is much more appropriate for many NLP applications, namely: ontology design by word clustering, word sense disambiguation, question-answering, pp-attachment, etc.

The main contribution of this paper is to define a protocol evaluation to compare the accuracy of our syntactic strategy against other window based techniques. Accuracy is defined with regard to the specific task of discovering word class relations. In order to perform this evaluation, cooccurrence data for different types of proper names (named entities), taking into account both syntactic dependencies and windowing relations, was collected from a Portuguese corpus of 12 million words. The corpus consists of articles of *O Público*, a general purpose Portuguese newspaper.

The paper is organized as follows: section 2 describes some related work. Then, section 3 briefly introduces two window-based methods, while section 4 describes more accurately our syntax based strategy, which relies on a very simple dependency parser. Finally, in Section 5, some experiments will be performed against a Portuguese corpus in order to evaluate the performance of the methods described in the previous sections.

2 Earlier Comparisons between both Approaches

Despite the growing interest in semantic extraction, there exist still few previous works aimed to evaluate and compare the two strategies at stake. In [6], a syntax-based method is carefully compared to a windowing technique, with regard to the general task of word similarity extraction. The former is shown to perform better for high-frequency words, while the windowing method is the better performer for low-frequency words. This evaluation was focused on associative links between words, since both methods were compared against online thesauri which are provided with all kind of semantic relations. So, we cannot know which method is more reliable for discovering cohyponymy relations between words. Moreover, the experiments performed made use of very small text corpora, probably due to the low efficiency of the syntactic techniques available at that time.

In [9], the two techniques are compared with regard to the extraction of several semantic relations. In most tasks, the syntax approach was clearly better than the bag-of-word model. However, the latter was defined in a very restrictive way: only the 200 most frequent words were considered as dimensions of context vectors. In [10], it is described a similar comparative experiment against a Dutch corpus. The authors conclude again that a full syntactic context model outperforms all other approaches.

[12] proposes a comparative evaluation of the two techniques with regard to a different task: extraction of multiwords and collocations. In the conclusion, they state that syntax-based methods outperform windowing techniques thanks to a drastic reduction of noise. The main problem is that experiments were performed with a parser which is not robust and time consuming (130 word/second [14]).

3 Window-Based Contexts

Contexts can be defined using the immediately adjacent words, within a window of n words. Two different techniques can be applied: one defining contexts as bag of words and the other taking into account word order.

The technique based on bag of words builds context vectors considering simple words as dimensions, regardless of their positions within the window. For instance, let's suppose that the Portuguese adjective *pequeno* ("small") cooccurs twice with *homem* ("man"): once to the left and once directly following the noun: *homem pequeno* and *pequeno homem*. Table 1 shows the contexts vectors of *homem* and *pequeno*. The value of each dimension is the number of cooccurrences without taking word position into account.

Table 1. Bag of words

	pequeno	*homem*
pequeno		2
homem	2	

Table 2. Word order

	$(p,<1)$	$(p,<2)$	$(p,>1)$	$(p,>2)$	$(h,<1)$	$(h,<2)$	$(h,>1)$	$(h,>2)$
pequeno					1		1	
homem	1		1					

On the other hand, Table 2 depicts the context vectors of the same two words when taking word order into account. Each dimension represents the position of the context word within a window of size 2. For instance, $(p,<1)$ means that *pequeno* is 1 word ahead of the main word. $(p,>2)$ represents two positions to the right. Using this technique, the vector size grows while frequency counting decreases. It results in a more sparse matrix. According to Rapp [11], word order

Table 3. Dependency triplets and patterns of POS tags

Dependencies	Patterns of POS tags
(green$_5$, $mod_<$, jacket$_6$) (big$_{10}$, $mod_<$, $d$dog$_{11}$)	*R$_1$: $s/(\mathbf{A_i})(\mathbf{N_j})/\mathbf{N_j}/$
()	*R$_2$: $s/(\mathbf{N_i})(\mathbf{N})_\mathbf{j}/\mathbf{N_i}/$
(man$_2$, $with_3$, jacket$_5$)	*R$_3$: $s/(\mathbf{N_i})(\mathbf{P_k})(\mathbf{N})_\mathbf{j}/\mathbf{N_i}/$
(see$_6$, $obj_>$, dog$_{11}$)	R$_4$: $s/(\mathbf{V_i})(?:D_k\vert R_n)*(\mathbf{N})_\mathbf{j}/\mathbf{V_i}/$
(see$_6$, $obj_<$, man$_2$)	R$_5$: $s/(?:D_k)*(\mathbf{N_i})(?:R_n)*(\mathbf{V})_\mathbf{j}/\mathbf{V_j}/$
()	R$_6$: $s/(\mathbf{V_i})(?:R_n)*(\mathbf{P_k})(?:\vert D_m\vert R_r)*(\mathbf{N})_\mathbf{j}/\mathbf{V_i}/$

is a statistical clue useful to simulate syntactic behavior. This window technique is, then, closer to the syntax-based approach.

4 Syntax-Based Contexts

The second technique to define word contexts relies on the identification of syntactic dependencies. So, context vectors will be provided with syntactic information.

4.1 Dependency Parsing with Generic Regular Expressions

Instead of searching for windows positions around words or lemmas, we make use of regular expressions to identify syntactic dependencies. Regular expressions represent basic patterns of POS tags which are supposed to stand for binary dependencies between two lemmas. Our parsing strategy consists of a sequence of syntactic rules, each rule being defined by a specific pattern of tags that stands for a binary dependency. This strategy is implemented as a finite-state cascade [1]. So far, our grammar is focused on dependencies with verbs, nouns, and adjectives, since it is assumed that these dependencies are useful for semantic extraction. Let's take an example. Suppose our corpus contains the following tagged sentence:

a_D$_1$ man_N$_2$ with_P$_3$ a_D$_4$ green_A$_5$ jacket_N$_6$ see_V$_7$ yesterday_R$_8$ a_D$_9$ big_A$_{10}$ dog_N$_{11}$

The aim is to identify dependencies between lemmas using basic patterns of POS tags. Dependencies are noted as triplets: $(head, rel, dependent)$. The first column of Table 3 shows the 5 triplets generated from the sentence above using the patterns appearing in the second column. Patterns are organized in a sequence of substitution rules in such a way that the input of a rule R_n is the output of a rule R_m, where $m \leq n$. A rule substitutes the POS tag of the head word (right side) for the whole pattern of tags representing the head-dependent relation (left side). The first rule, R_1, takes as input a string containing the ordered list of all tags in the sentence:

$D_1N_2P_3D_4A_5N_6V_7R_8D_9A_{10}N_{11}$

The left pattern in this rule identifies two specific adjective-noun dependencies, namely "A_5N_6" and "$A_{10}N_{11}$". As a result, it removes the two adjective tags from the input list, and produces as output:

$D_1N_2P_3D_4N_6V_7R_8D_9N_{11}$

Then, rule R_3 is applied to the output of R_1. The left pattern of this rule matches "$N_2P_3D_4A_5N_6$" and rewrites the following ordered list of tags:

$D_1N_2V_7R_8D_9N_{11}$

This list is the input of the following applicable rule, R_4, which produces:

$D_1N_2V_7$

Finally, rule R_5 is applied and gives as result only one tag, V_7, which is associated to the root head of the sentence: the verb "see". As this verb does not modify any word, no rule can be applied and the process stops. This is in accordance with the main assumption of dependency-based analysis, namely, a word in the sentence may have several modifiers, but each word may modify at most one word [8]. In sum, each application of a rule, not only rewrites a new version of the list of tags, but also generates the corresponding dependency triplet. So, even if we do not get the correct root head at the end of the analysis, the parser generates as many triplets as possible. This strategy can be seen as a particular case of partial and robust parsing [1], which is as faster as identifying contextual words with a window-based technique (over 7000 words/second).

The 5 triplets in Table 3 where generated from 4 substitution rules, each matching a type of dependency: adjective-noun, noun-prep-noun, verb-noun, and noun-verb. The sentence analyzed above does not contain triplets instantiating noun-noun and verb-prep-noun dependencies. Wildcards $(?: D|R)*$ stand for optional determiners and adverbs, that is, they represent optional sequences of determiners or/and adverbs that are not considered for triplets. Rules with an asterisk can be applied several times before applying the next rule (e.g., when a noun is modified by several adjectives). Subscript numbers allow us to link tags in the patterns with their corresponding lemmas in the sentence. To represent triplets, we use 4 types of binary relations: prepositions, left modifiers (noted as $mod_<$), right objects ($obj_>$), and left objects ($obj_<$). Note that the patterns of tags in Table 3 work well with English texts, but they are so generic that they also can be used for many languages. To extract triplets from texts in Romance languages such as Portuguese, Spanish, French, or Galician, 2 tiny changes are required: to provide a new pattern with dependent adjectives at the right position of nouns ($mod_>$), and to take as the head of a noun-noun dependency the noun appearing at the left position. The experiments that will be described later were performed over a Portuguese corpus. To date, our parser can be applied on text previously tagged with either Treetagger[1] and Freeling [2].

[1] *http://www.ims.uni-stuttgart.de/projekte/corplex/Tree-Tagger/DecisionTreeTagger.html*

4.2 Lexico-Syntactic Contexts

The second step of our syntax-based method consists in extracting lexico-syntactic contexts from the dependencies and counting the occurrences of lemmas in those contexts. This information is stored in a collocation database. The extracted triplets of our example allow us to easily build the collocation database depicted in Table 4. The first line of the table describes the entry "man". This noun occurs once in two lexico-syntactic contexts, namely that representing the left position ($obj_<$) of the verb "see", $(\text{see}, obj_<, N)$, and that denoting the noun position being modified by the prepositional complement "with a jacket". The second line describes the entry "see", which also occurs once in two different lexico-syntactic contexts: $(V, obj_<, man)$ and $(V, obj_>, dog)$, i.e., it co-occurs with both a left object, "man", and a right object: "dog". The remaining lines describe the collocation information of the remaining nouns and adjectives appearing in the sentence above.

Table 4. Collocation database of lemmas and lexico-syntactic contexts

Lemmas	Lexico-Syntactic Patterns and freqs.
man	$< (\text{see}, obj_<, N), 1 >$ $< (N, with, \text{jacket}), 1 >$
see	$< (V, obj_<, \text{man}), 1 >$ $< (V, obj_>, \text{dog}), 1 >$
big	$< (\text{dog}, mod_<, A), 1 >$
dog	$< (N, mod_<, \text{big}), 1 >$ $< (\text{see}, obj_>, N), 1 >$
green	$< (\text{jacket}, mod_<, A), 1 >$
jacket	$< (N, mod_<, \text{green}), 1 >$ $< (\text{man}, with, N), 1 >$

Notice we always extract 2 complementary lexico-syntactic contexts from a triplet. For instance, from $(\text{man}, with, \text{jacket})$, we extract:
$(N, with, \text{jacket})$ $(\text{man}, with, N)$
This is in accordance with the notion of co-requirement defined in [5]. In this work, two syntactically dependent words are no longer interpreted as a standard "predicate-argument" structure, where the predicate is the active function imposing syntactic and semantic conditions on a passive argument, which matches such conditions. On the contrary, each word in a binary dependency is perceived simultaneously as a predicate and an argument. In the example above, $(\text{man}, with, N)$ is seen as an unary predicate that requires nouns denoting parts of men (e.g. jackets), and simultaneously, $(N, with, \text{jacket})$ is another unary predicate requiring entities having jackets (e.g. men).

Finally, syntax-based context vectors are easily built from the collocation database. As in [7] and [5], we use several types of dependencies to define syntactic contexts, and not only objects and subjects.

5 Experiments

5.1 Corpus

Experiments have been carried out using a Portuguese corpus with 12 million tokens extracted from the general-purpose journal *O Público*. Before building the window and syntax based contexts, texts were lemmatized and *POS* tagged with TreeTagger.[2]. In the case of window contexts (both bag of words and word order), function words were previously removed.

5.2 Vector Similarity

The similarity coefficient used in our experiments to compare vector contexts is a particular version of Dice score: *dice*†. Similarity between the context vectors of two lemmas, $lemma_1$ and $lemma_2$, is computed as follows:

$$dice\dagger(lemma_1, lemma_2) = \frac{2 * \sum_i min(f(lemma_1, cntx_i), f(lemma_2, cntx_i))}{F(lemma_1) + F(cntx_2)}$$

where $f(lemma_1, cntx_i)$ represents the number of times $lemma_1$ cooccurs with $cntx_i$. $F(lemma_i)$ stands for the absolute frequency of $lemma_1$. We use this coefficient because it produced the best results in related work [3,13,4].

5.3 Initial List of Seed Proper Nouns

Our objective is to design an evaluation protocol avoiding unclear and fuzzy judgments about word similarity. For this purpose, we only consider a reduced sample list of proper nouns. For each member of the list, we compute its *dice*† similarity with all proper nouns in the corpus, and produce a ranked list with its top 5 most similar nouns. The test list was built by hand and consists of 28 proper nouns divided in 7 semantic categories: countries, capitals of countries, Portuguese towns, politicians, organizations, press agencies, and football teams. As we selected 5 similar candidates for each test noun, the final list to be evaluated contains 140 proper nouns. The evaluator is just required to classify each new proper name as a member of the 7 categories enumerated above. For instance, if *Washington* is selected as a similar noun to *Bruxelas*, which belongs to the category of capitals within the test list, the evaluator only needs to decide if *Washington* is or not a capital.

Furthermore, the 28 test proper nouns were selected according to their position in the list of all nouns ranked by frequency. They were required to be distributed in 4 ranges: (1) very frequent words, ranked between 1 and 1000, with frequency > 479; (2) quite frequent words, ranked between $1,000$ and $3,000$, and whose frequency is $> 100 < 479$; (3) not very frequent words, ranked between $3,000$ and $5,000$, with frequency $> 50 < 100$; (4) quite rare words, ranked between $5,000$ and $10,000$, with frequency $> 20 < 50$.

[2] For Portuguese, see in *http://gramatica.usc.es/∼gamallo/tagger.htm*

5.4 Results

We measured the precision of three methods: two window based techniques, one relying on bag of words and the other on word order, and our syntax based method. For each method, we computed the total precision and that obtained for each one of the 4 frequency ranges considered.

Table 5 shows the list of candidates obtained from 3 test proper nouns: the first one is *Peru*, designing a country, and situated in range (3), that is, its frequency in the corpus is $> 50 < 100$. The second one is a capital, *Belgrado*, also situated in range (3), and finally, a Portuguese town, *Viseu*, situated in range (2). Correct candidates are in bold.

Table 5. Similar words according to the 3 methods

name	**bag of words**	**word order**	**syntax**
Peru	Alberto Fujimori, PRESIDENTE, Abimael, Vargas Llosa	Resultados, Política, Humanidade, Fernando Sousa, **Argentina**	**Tchetchénia**, Sul de Espanha, **Guiné-Bissau, Libéria, Guatemala**
Belgrado	SPS, EF, Solidariedade, Pedro Caldeira Rodrigues, Unidades	SPS, Líbia, **Krajina**, Sérbia, **Jacarta**	**Moscovo, Washington, Jacarta, Zagreb, Argel**
Viseu	Juventude, Pereira, **Montijo**, Teatro, **Guarda**	Intervenção, Olivieria de Azeméis, **Guarda**, Australia, **Castelo Branco**	**Bragança, Beja, Guarda, Santarém, Leiria**

Notice that the method based on bag of words does not select any country from *Peru*, but is able to retrieve two individuals associated to this country: both *Alberto Fujimori* and *Vargas Llosa*. This is in accordance with one of our initial assumptions: window-based techniques are not suitable to extract word class relations (co-hyponymy), but rather any kind of associative link between words.

Table 6 depicts results on precision for the 3 methods, taking into account the 4 frequency ranges of lemmas as well as all lemmas with frequency > 20. The results show that the syntax-based method performs much better than the windowing techniques, whereas the strategy based on word order is quite better than that relying on just bag of words (see Figure 2). So, structural information (dependencies and word order) helps identifying meaningful contexts. On the other hand, the former method is the only one that clearly improves when it is being applied on more and more frequent lemmas (see Figure 1). Hence, it follows that the syntactic strategy would perform better as corpus size grows, which is not true for the two windowing techniques. This is somehow in accordance with

Table 6. Results of the 3 evaluated methods

Range / freq	**Syntax** $117,266$ cntxs **Prec-%**	**Word order** $190,228$ cntxs **Prec-%**	**Bag of words** $148,422$ cntxs **Prec-%**
1-1000 > 450	97	63	29
1000-3000 $> 100 < 450$	97	49	37
3000-5000 $> 50 < 100$	80	57	31
5000-10000 $> 20 < 50$	57	54	22
Total > 20	83	55	36

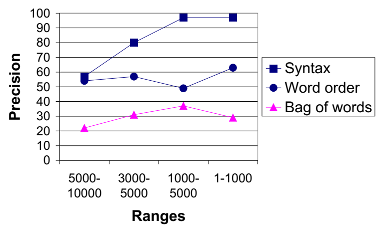

Fig. 1. Precision scores by ranges of frequency

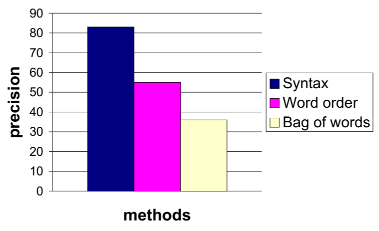

Fig. 2. Precision scores for proper nouns with frequency > 20

the experiments performed by Grefenstette [6], where the window-based method was the better performer for low-frequency words.

In Table 6, we also show the number of syntactic contexts used by each method. Let's note that the number of syntactic contexts $(117,226)$) is much smaller than that of window based contexts. As the size of context vectors in the syntactic approach is not very large, the process of computing similarities turns out to be more efficient.

6 Conclusion

We consider that syntactic analysis of source corpora is more suitable for extraction of co-hyponymy semantic relationships, and that the syntactic structure of source text has to be taken into account in order to ensure the quality of results for both frequent and not frequent words. In addition, our syntax-based method is more computationally efficient than the windowing techniques since it defines and uses smaller context vectors. On the other hand, the syntactic strategy defined in this paper can be considered as knowledge-poor as the window-based approach, since the robust parsing described here relies on few generic regular expressions. Moreover, as the generic knowledge underlying the parser is used to identify basic dependencies for several natural languages, our multilingual strategy turns out to be almost as language-independent as any windowing technique. In sum, in order to extract co-hyponymy, it seems to us there are no strong arguments to use window techniques instead of syntactic contexts.

References

1. Steven Abney. Part-of-speech tagging and partial parsing. In Ken Church, Steve Young, and Gerrit Bloothooft, editors, *Corpus-Based Methods in Language and Speech.* Kluwer Academic Publishers, Dordrecht, 1996.
2. X. Carreras, I. Chao, L. Padró, and M. Padró. An open-source suite of language analyzers. In *LREC'04*, Lisbon, Portugal, 2004.
3. James R. Curran and Marc Moens. Improvements in automatic thesaurus extraction. In *ACL Workshop on Unsupervised Lexical Acquisition*, pages 59–66, Philadelphia, 2002.
4. Pablo Gamallo. Learning bilingual lexicons from comparable english and spanish corpora. In *Machine Translation SUMMIT XI*, Copenhagen, Denmark, 2007.
5. Pablo Gamallo, Alexandre Agustini, and Gabriel Lopes. Clustering syntactic positions with similar semantic requirements. *Computational Linguistics*, 31(1):107–146, 2005.
6. Gregory Grefenstette. Evaluation techniques for automatic semantic extraction: Comparing syntactic and window-based approaches. In *Workshop on Acquisition of Lexical Knowledge from Text SIGLEX/ACL*, Columbus, OH, 1993.
7. Dekang Lin. Automatic retrieval and clustering of similar words. In *COLING-ACL'98*, Montreal, 1998.
8. Dekang Lin. Dependency-based evaluation of minipar. In *Workshop on Evaluation of Parsing Systems*, Granada, Spain, 1998.
9. Sebastian Padó and Mirella Lapata. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199, 2007.
10. Yves Peirsman, Kris Heylen, and Dirk Speelman. Finding semantically related words in dutch. co-occurrences versus syntactic contexts. In *CoSMO Workshop*, pages 9–16, Roskilde, Denmark, 2007.
11. Reinhard Rapp. Automatic identification of word translations from unrelated english and german corpora. In *ACL'99*, pages 519–526, 1999.
12. Violeta Seretan and Eric Wehrli. Accurate collocation extraction using a multilingual parser. In *COLING-ACL'06*, pages 953–960, 2006.
13. Lonneke van der Plas and Gosse Bouma. Syntactic contexts for finding semantically related words. In *CLIN2004*, 2004.
14. Eric wehrli. Fips, a deep linguistic multilingual parser. In *5th Workshop on Important Unresolved Matters*, pages 120–127, 2005.