An Improved Automatic Term Recognition Method for Spanish

Alberto Barrón-Cedeño[1,2], Gerardo Sierra[1], Patrick Drouin[3], and Sophia Ananiadou[4]

[1] Engineering Institute, Universidad Nacional Autónoma de México, Mexico
[2] Department of Information Systems and Computation, Universidad Politécnica de Valencia, Spain
[3] Observatoire de Linguistique Sense-Texte, Université de Montréal, Canada
[4] University of Manchester and National Centre for Text Mining, UK
alberto@pumas.ii.unam.mx, gsierram@ii.unam.mx, patrick.drouin@umontreal.ca, sophia.ananiadou@manchester.ac.uk

Abstract. The *C-value/NC-value* algorithm, a hybrid approach to automatic term recognition, has been originally developed to extract multiword term candidates from specialised documents written in English. Here, we present three main modifications to this algorithm that affect how the obtained output is refined. The first modification aims to maximise the number of real terms in the list of candidates with a new approach for the stop-list application process. The second modification adapts the *C-value* calculation formula in order to consider single word terms. The third modification changes how the term candidates are grouped, exploiting a lemmatised version of the input corpus. Additionally, size of candidate's context window is variable. We also show the necessary linguistic modifications to apply this algorithm to the recognition of term candidates in Spanish.

1 Introduction

The *C-value/NC-value* algorithm [3] is the base of the Termine suite for automatic multiword terms recognition in specialised documents in English[1]. TerMine is a text mining service developed by the UK National Centre for Text Mining for the automatic extraction of terms in a variety of domains. This algorithm has been applied to Automatic Term Recognition (ATR) over different languages such as English [3] and Japanese [10]. Additionally, it is the base for an algorithm designed for term extraction in Chinese [4]. A first essay has started to adapt it to handle documents in Spanish [1].

In this paper, we describe the improvements carried out over different stages of the algorithm. Additionally, we show the necessary adaptations for exploiting this algorithm on ATR of terms in Spanish texts. About the distribution of

[1] http://www.nactem.ac.uk/software/termine/

the paper, Section 2 gives a brief description of the original algorithm. Section 3 describes the corpora exploited during the design and test of our method. Section 4 describes the modifications we have made to the algorithm including a description of the resources we have exploited. Section 5 contains the evaluations. Finally, Section 6 draws some conclusions and future work.

2 The *C-value/NC-value* Algorithm

The *C-value/NC-value* algorithm was originally developed by Frantzi et al. [3] for multiword ATR on English texts. This hybrid (linguistic-statistical) algorithm is divided into two main stages: *C-value* and *NC-value*. We summarise below the main ideas developed in that paper. (Subsections 2.1 and 2.2, respectively).

2.1 *C-value*: the Hybrid Stage

The task of the *C-value* algorithm is to process an input corpus (composed of a set of specialised texts) in order to generate a list of candidate terms. This list is ranked according to the potential of each candidate of being a real term: its *termhood*.

Figure 1 shows a simplified version of the *C-value* algorithm. The entire version can be found in [3]. The linguistic steps that we have modified will be described in Section 4. In our current approach, we have not defined any threshold in order to previously remove strings (fourth line). The threshold exceeding conditions that let us decide if the *C-value* of a candidate is high enough to consider it as a good term candidate are not used either. The reason is that we do not want to discriminate candidates based on their frequency or *C-value* because, until now, our corpus is not big enough to reach a significant threshold.

The Linguistic Stage. The linguistic filter recognises noun phrases (combination of nouns with prepositions and adjectives), which are potential terms. Section 5.1 contains a comparison of the results obtained by using an open versus a closed filter.

The stop-list is composed of a set of words which are not expected to occur inside of a term on the studied domain (due to this fact the stop-list is domain-dependent). Those candidates with words in the stop-list are removed from the list (Section 4.2 shows the improvement made to this filtering process). The list of candidates obtained by this linguistic process is ranked on the basis of the next statistical process.

The Statistical Stage. The purpose of this part of the algorithm is to measure the termhood of every candidate string. The list of candidate terms is ranked based on this value (*C-value*). The *C-value* calculation considers four aspects:

1. The frequency of the candidate in the entire corpus.
2. The frequency of the candidate when it appears nested in longer candidates.
3. The number of those longer candidates.
4. The length of the candidate (in words).

Algorithm 1: Given the analysis corpus:

```
$outputList = [\,]$
tag the corpus
extract strings using linguistic filter
remove strings below frequency threshold
filter rest of strings through stop-list
For each string $a$ given that $length(a) = max$
    $C\text{-}value(a) = log_2|a| * f(a)$
    If $C\text{-}value(a) \geq Threshold$
      add $a$ to $outputList$
      For each substring $b \in a$
          revise $t(b)$ and $c(b)$
For each string $a$ given that $length(a) < max$
    If $a$ appears for the first time
      $C\text{-}value(a) = log_2|a| * f(a)$
    Else
      $C\text{-}value(a) = log_2|a|(f(a) - \frac{1}{c(a)}t(a))$
    If $C\text{-}value(a) \geq Threshold$
      add $a$ to $outputList$
      For each substring $b \in a$
          revise $t(b)$ and $c(b)$
```

Fig. 1. Simplified *C-value* algorithm

The absolute frequency of a term candidate in a corpus is an initial parameter to define if it is a real term. However, it is not enough. In fact, it is common that long terms appear just once even in long corpora. After this appearance, references to this kind of terms often appear only as truncated versions of themself.

For example, in a sample of 139,027 words from the Computer Science Corpus in Spanish [7], the term *computadora* (computer) appeared 122 times. Meanwhile, *computadora personal* (personal computer) appeared only 15 times. In this case, we could guess that, at least in some cases, *computadora* is a simplified version of *computadora personal*, so the nested appearance of the former in the latter decreases the probability of *computadora* of being a real term.

Nevertheless, in the same sample corpus there are some other candidates that are built having to the string *computadora* as the root, such as *computadora portátil* (laptop) and *computadora de uso general* (general purpose computer) appearing 15 and 2 times, respectively. These three different term candidates with the same root, reflect the possibility of *computadora* of being by itself a real term. The other three strings could be varieties with its own concept associated in the subject area (as it is). In such a case, it is considered that all four candidates could be real terms. For these reasons, three considerations are made:

1. The high frequency of a candidate in a corpus is beneficial for its termhood.
2. The length of a string is also beneficial (due to the fact that the probability of a long string of appearing in a corpus decreases as it is longer).

3. The appearance of a candidate nested into another detriments its termhood.
4. If a candidate appears nested in multiple candidate strings, the detrimental effect becomes weaker.

The *C-value* is calculated as in Eq. 1.

$$C\text{-}value = \begin{cases} log_2|a| * f(a) & \text{if } a \notin nested \\ log_2|a| * \left(f(a) - \frac{1}{P(T_a)} \sum_{b \in T} f(b)\right) & \text{otherwise} \end{cases} , \tag{1}$$

where a is the candidate string, $f(\cdot)$ is the frequency of the string $\cdot$ in the corpus, T_a is the set of extracted candidates containing a, and $P(T_a)$ is the number of those candidates. The *nested* set is composed of all those candidates appearing inside of longer candidates.

This process generates the list T_1 of term candidates. T_1 contains the set of candidates ranked by their termhood (*C-value*).

2.2 *NC-value*: Considering the Terms Context

It is hard to think in a word without relating it to some others that interact with it. Sager [11] has stated that terms tend to be accompanied by a strict set of other words (including more terms). In order to illustrate, consider the term *hard disk*. This term will hardly appear with words such as *cook* on its neighbourhood, but it will frequently appear with words such as *GB*, *format*, *install* or *capacity*, which are related to it.

If a term appears with a "closed" set of neighbour words, the existence of these words in the context of a candidate must be positive clues for its termhood. The *NC-value* method extends *C-value* by considering the candidates context with the so called *context weighting factor*. Term context words are those appearing in the neighbourhood of the candidates. However, not all the words in the neighbourhood must be considered as context words. Only nouns, adjectives and verbs (other words do not add significant information to a term).

A list of obtained context words is obtained and ranked according to their "relevance" over the terms. This relevance is based on the number of terms that appear in their contexts. The higher this number, the higher the probability that the word is related to real terms. It is expected that these words appear with other terms in the same corpus. The context weighting factor (that expresses the probability of a word w of being a term context word) is calculated as in Eq. 2.

$$weight(w) = \frac{t(w)}{n} , \tag{2}$$

where w is a term context word, $weight(w)$ is the weight assigned to the word w (expressed as a probability), $t(w)$ is the number of terms w appears with, and n is the total number of terms considered.

The weight assigned to a context words must be calculated after getting the list T_1 that, as we have said, is ordered by the *C-value*. In order to extract the term context words, the top candidate terms in T_1, which present a high

Precision (contains a good proportion of real terms), is used. These top terms produce a list of term context words weighted on the basis of Eq. 2.

The rest of the context words may or may not have an associated context weight. In the case where a context word w has been seen earlier, it retains its associated weight. Otherwise, $weight(w) = 0$. The *NC-value* for the term candidates is calculates as in Eq. 3, which considers the previously calculated *C-value* as well as the context words weights:

$$NC\text{-}value = 0.8C\text{-}value(a) + 0.2 \sum_{b \in C_a} f_a(b) weight(b), \tag{3}$$

where a is the current term candidate, C_a is the set of context words associated to a, b is each one of those context words, and $f_a(b)$ is the frequency of b as a context word of a.

The context information is exploited in order to concentrate the real terms in the top of the list. The new list T_2 of term candidates is ranked on the basis of the *NC-value*.

3 The Corpora

We have used two corpora during the design and evaluation of our prototype: The Linguistic Corpus of Engineering and the Computer Science Corpus in Spanish, described in Subsections 3.1 and 3.2, respectively.

3.1 The Linguistic Corpus of Engineering

The Linguistic Corpus of Engineering (CLI) [9] has been created in the Language Engineering Group at the Universidad Nacional Autónoma de México (UNAM). It is composed of a set of specialised texts on Engineering (mechanics, civil and electronics, among others). Most of the documents are written in Mexican Spanish and includes some texts written in peninsular Spanish. This corpus, consisting of 23 files with 274,672 words, includes postgraduate as well as undergraduate thesis, papers and reports on this area.

Due to the fact that Engineering is a large subject area, we have opted for focusing only on the CLI Mechanical Engineering section. This section includes 5 files for a total of 10,191 words.

The CLI corpus has been used in order to define the rules for the linguistic filter definition corresponding to the candidate extraction subtask (Subsection 4.1).

3.2 The Computer Science Corpus in Spanish

The Computer Science Corpus in Spanish (CSCS) [7] was compiled in the *Observatoire Linguistique Sense-Texte* (University of Montreal). The original objective of this corpus is the development of a Spanish version of DicoInfo [5] “the fundamental computer science and Internet dictionary”[2].

[2] http://olst.ling.umontreal.ca/dicoinfo/

Table 1. Statistics of the CLI and CSCS corpora sections used in our experiments

Feature	Value
CLI	
Number of files	5
Total number of tokens	10,191
Avg. number of tokens per file	2,038
CSCS	
Number of files	200
Total number of tokens	150,000
Avg. number of tokens per file	750

The CSCS contains around 550 documents with more than 500,000 words. It mainly contains texts written in peninsular Spanish. For our experiment, we have chosen the Hardware section, with around 200 documents and almost 150,000 words.

This corpus has been used in order to define the open linguistic filter, corresponding to the candidate extraction task (Subsection 4.1), as well as for evaluation (Subsection 5.1). Some statistics for both corpora are included in Table 1.

4 Improvements to the Algorithm

After explaining the original *C-value/NC-value* algorithm, as well as describing the used corpora, we discuss the adaptations carried out to both the linguistic and statistical sections of the *C-value* method.

4.1 Creating the Linguistic Filter for Spanish

The modified prototype has been designed for ATR over documents written in Spanish. For our experiments we have considered the application of two filters: closed and open. The first one is strict and tries to retrieve only real terms, reducing the number of false positives. The latter is flexible and tries to retrieve all the terms in the corpus no matter the number of false negatives obtained.

The most frequent term patterns in Spanish are Noun -*amplificador*, *protocolo* (amplifier, protocol)-, Noun Prep Noun -*estación de trabajo*, *lenguaje de programación* (work station, programming language)- and Noun Adjective -*computadora personal*, *red neuronal* (personal computer, neural network)- [2]. These patterns compose our closed filter (this set of rules as well as the corresponding to the open filter are in NLTK format [8]):

- $NounAdj$
- $NounPrepDEAdj$
- $Noun$

In the second rule we do not consider any preposition, but only *de* (of). That is the meaning of the tag *PrepDE*.

Additionally, we have carried out a manual term extraction based on the method described in [6]. The objective was to find more flexible patterns in order to retrieve more terms (while trying to limit the generation of noise in the output). The manual extraction carried out over a section of both, CLI and CSCS, corpora resulted in the following set of rules composing the open filter:

- $(Noun|ProperNoun|ForeignWord)^+$
- $(NounAdj)(PrepDE(Noun|ProperNoun))^*$
- $NounPrepDE(Noun|ProperNoun)$
- $Noun^? Acrnm$
- $NounPrepDE((NounAdj)|(AdjNoun))$

Note that some terms contained foreign words (most of them in English). Other part-of-speech such as acronyms and proper nouns have appeared also. The closed or open filter depends on the interest of favouring Precision or Recall in the output (Section 5).

4.2 Modifications to the *C-value* Algorithm

We have detected some weaknesses to the *C-value/NC-value* algorithm. With the aim of reducing them, we have carried out four main modifications.

Selective Stop-Words Deletion. As we have pointed out in Subsection 2.1, a stop-list is applied during the *C-value* stage in order to reduce noise. The original method deletes an entire candidate if it contains at least one stop-word.

A stop-list in ATR is a list of words which are not expected to occur as term words in the treated domain. Our stop-list contains 223 words. It is composed of nouns and adjectives that presented a high frequency in the CSCS but it is not expected to find them inside of real terms. Some examples of these stop-words are *característica*, *compañía* and *tamaño* (feature, company and size, respectively).

Our strategy propose the deletion of stop-words instead of entire candidates. We call this strategy *selective stop-words deletion*. The reason for this selective deletion is that there are a lot of candidate terms containing stop as well as other kind of words. For instance, consider the candidate *computadora grande* (big computer). If we only delete the substring *grande* instead of the entire candidate, keeping *computadora*, the patterm of the obtained candidate is characteristic of the terms in Spanish. And, as it is the case in Computer Science, it becomes a potential real term.

However, the stop-words could be linked to functional words. In order to clarify this point, consider another example. The candidate *desarrollo de LCD* (LCD's development) contains the stop-word *desarrollo*. The POS of this candidate is *Noun PrepDE Noun*. Again, the basic option would be completely discarding this candidate, but *LCD* is a real term. On the selective stop-words deletion strategy, we only delete the stop-word (*desarrollo*). On this way, we obtain *de LCD* with POS *prepDE Noun*. However, this is not a characteristic pattern of terms in Spanish. If the stop-word is a noun, it is necessary to check those words before and after it in order to decide if they should be deleted also. In this case, the preposition *de* must be deleted. The result is *LCD*, which is a real term.

The selective stop-words deletion strategy is described in Algorithm 2. This algorithm has been designed for Spanish terms. However, after a brief linguistic adaptation it is possible to apply it to any other language).

Algorithm 2: Given a candidate s split into words s_i:

```
$D = \{\}$ //The set of words that will be deleted from $s$
If $\mathcal{P}(s_i) = Adjective$ and $s_i \in$ stop-list
  add $s_i$ to $D$
Elif $\mathcal{P}(s_i) = Noun$ and $s_i \in$ stop-list
  add $s_i$ to $D$
  If $\mathcal{P}(s_{i-1}) = Preposition$
    add $s_{i-1}$ to $D$
  If $\mathcal{P}(s_{i+1}) = Preposition$ or $\mathcal{P}(s_{i+1}) = Adjective$
    add $s_{i+1}$ to $D$
    If $\mathcal{P}(s_{i+2}) = Preposition$ or $\mathcal{P}(s_{i+2}) = Adjective$
      add $s_{i+2}$ to $D$
delete words in $D$ from $s$
Return $s$
```

Fig. 2. Selective deletion of stop and related words $\mathcal{P}(\cdot)$ = part-of-speech of $\cdot$

In order to clarify how Algorithm 2 works, we give another example. Consider the candidate *pantalla de manera lateral* (screen in lateral way). In this case, $s = \{pantalla_{<Noun>}, de_{<PrepDE>}, manera_{<Noun>}, lateral_{<Adj>}\}$. s_2 (manera) is a stop-word, so $D = \{s_2\}$. *manera* is a noun and for this reason it is necessary to check s_{2-1}, which is a preposition. In this step $D = \{s_1, s_2\}$. The word s_{2+1} (an adjective) must be deleted. Now $D = \{s_1, s_2, s_3\}$. The resulting candidate after deleting D from s is *pantalla* (screen).

Modifying the *C-value* Calculation Formula. The *C-value/NC-value* algorithm was originally designed for the extraction of multiword terms. It is for this reason that the *C-value* calculation formula was not designed to handle terms composed of one word.

Fortunately, this limit is only mathematical. The *C-value* formula is not able to calculate termhood for candidates with $length(a) = 1$ since, in order to normalise the length relevance, it calculates its logarithm (note that $log(1) = 0$, so $C\text{-}value(a) = 0$). In order to avoid this limitation, we add a constant i to the length of a before calculating its logarithm:

$$C\text{-}value = \begin{cases} c * f(a) & \text{if } a \notin nested \\ c * \left(f(a) - \frac{1}{P(T_a)} \sum_{b \in T} f(b) \right) & \text{otherwise} \end{cases}, \quad (4)$$

where $c = i + log_2|a|$.

On the initial experiments, we tried $i = 0.1$ in order to modify as less as possible the essence of the formula. However, real terms with $length = 1$ used to appear too far in the bottom of the output list, after a lot of bad longer candidates. It is for this reason that reason we define $i = 1$, which (experimentally) produces better rankings.

Searching on a Lemmatised Corpus. The first step in the *C-/NC-value* algorithm is POS tagging the corpus. However, the example output included in [3], includes the strings *B cell* and *B cells* in different rows of the *NC-value* ranked list. This reflects that there is no lemmatisation process involved. We consider that including such a process is important for term extraction. In the case when the lemmatised corpus is used to build the candidate terms list, different variations of the same candidate term are considered as one and its total frequency is the addition of all the variations frequencies.

In order to join the different variations of a candidate, we lemmatise the corpus before processing it. We carry out this subtask with TreeTagger [12]. This tool is a POS tagger as well as a lemmatiser.

4.3 Modifying the *NC-value* Algorithm

The *NC-value* stage is based on considering the candidates context. A word appearing frequently in the neighbourhood of a term in the top of the list ranked by *C-value* (with a high probability of being a real term) has a good probability of appearing with other real terms (no matter if they are in a lower position of the list).

The context for the candidates was originally defined as a fixed window of length 5. However, we have opted for using flexible frontiers to define the context windows. Punctuation marks (point, colon, semicolon, parenthesis) break phrases. Due to this fact a context window is broken if it contains one of these marks (no matter the length of the resulting window).

5 Evaluation

Our version of the *C-value/NC-value* algorithm for ATR has been evaluated in terms of Precision and Recall. In 5.1 we evaluate the extractor with different configuration parameters. Section 5.2 compares our adaptation to another previously designed for Chinese [4].

5.1 Varying the Parameters for the Extraction

We have randomly selected a set of documents from the CSCS [7]. The test corpus contains 15,992 words on the Hardware subject. In order to evaluate the obtained results we have carried out a manual term extraction over the same test corpus.

We have carried four experiments in order to compare different parameters combinations. These combinations are the following:

A Open linguistic filter without stop-list
B Open linguistic filter with stop-list
C Closed linguistic filter without stop-list
D Closed linguistic filter with stop-list

The open and closed filters are described in section 4.1. An open linguistic filter is flexible with the terms patterns. For this reason, it increases Recall reducing Precision. A closed linguistic filter is strict with the accepted patterns. For this reason, it increases Precision reducing Recall.

A total of 520 terms were found during the manual extraction process. The results obtained by the different automatic extractions are shown in Table 2.

Table 2. Result of automatic extractions with different parameters

Case	**Candidates**	**Real terms**	**P**	**R**
A	1,867	430	0.230	0.826
B	1,554	413	0.265	0.794
C	1,000	241	0.240	0.463
D	850	262	0.308	0.503

As it is expected, considering an open filter benefits Recall but harms Precision while the using a closed filter benefits precision but harms Recall. Looking more closely at the results obtained by experiments C and D, we can see that the Recall obtained by the latter is higher. This improvement is due to the fact that after the selective deletion, carried out in experiment D, more real terms (mainly of length 1) that originally appeared combined to stop-words are discovered. The original approach discards those candidates (Section 4.2).

5.2 Comparing our Adaptation with a Chinese Version

The *C-value/NC-value* method has been implemented and modified previously. [4] have developed a term extractor for texts on IT written in Chinese. In this case, the *NC-value* stage is replaced by a semantic and syntactic analysis stage. The objective of this stage is better ranking the obtained output.

The reported experiments on a sample corpus of 16 papers (1,500,000 Chinese characters) obtain $Precision = 0.67$ and $Recall = 0.42$. Our experiment B, obtains $Precision = 0.265$ and $Recall = 0.794$. Although their Precision is better than ours, we must consider that they use a previously obtained list of 288,000 terms of $length = 1$. This list is a filter that separates good candidates from bad ones.

Unlike them, we have opted for conserving the philosophy of the *C-value/ NC-value* method: our approach only needs a POS tagger in order to carry out the extraction process.

We must say that we have not compared our algorithm to the originally described in [3], because our experiment conditions are quite different mainly from the stop-list and the corpus features points of view.

6 Conclusions

In this paper we have described a linguistic and functional adaptation of the *C-value/NC-value* algorithm for automatic term recognition. The main functional adaptations carried out are the following:

– A new algorithm for the selective elimination of stop-words in the term candidates has been designed.
– The *C-value* calculation formula has been adapted in order to allow handle candidates of one word.
– The length of the candidates context windows has been is not fixed. Unlike the default $length = 5$, it is dynamically re-sized when it includes punctuation marks.

About the linguistic adaptations, we have analysed the patterns of the terms in Spanish in order to build an open and a closed filter for candidates detection. The open filter favours Recall, while the closed filter favours Precision. Additionally a stop-list composed of around 200 nouns and adjectives has been created.

With respect to other versions of *C-value/NC-value* method, our obtained Precision has decreased. The main reason for this behaviour is that we consider candidates of one word. Moreover, we have not defined any threshold in order to eliminate candidates with low frequency or *C-value*. We have opted for supporting the noise for the sake of a minimum loss of information, resulting in a good Recall.

Finally, we have designed a selective stop-words deletion method. Our method discovers good term candidates that are ignored when considering the original stop-word deletion method.

Acknowledgements. This paper has been partially supported by the National Council for Science and Technology (CONACYT); the DGAPA-UNAM; the General Direction of Postgraduate Studies (DGEP), UNAM; and the Macro-Project *Tecnologías para la Universidad de la Información y la Computación*, UNAM.

References

1. Barrón, A., Sierra, G., Villaseñor, E.: C-value aplicado a la extracción de términos multipalabra en documentos técnicos y científicos en español. In: 7th Mexican International Conference on Computer Science (ENC 2006). IEEE Computer Press, Los Alamitos (2006)
2. Cardero, A.M.: Terminología y Procesamiento. Universidad Nacional Autónoma de México, Mexico (2003)
3. Frantzi, K., Ananiadou, S., Mima, H.: Automatic recognition of multi-word terms: the C-value/NC-value method. International Journal on Digital Libraries 3(2), 115–130 (2000)

4. Ji, L., Sum, M., Lu, Q., Li, W., Chen, Y.-R.: Chinese terminology extraction using window-based contextual information. In: Gelbukh, A. (ed.) CICLing 2007. LNCS, vol. 4394, pp. 62–74. Springer, Heidelberg (2007)
5. L'Homme, M.C.: Conception d'un dictionarie fundamental de l'informatique et de l'Internet: selection des entrées. Le langage et l'homme 40(1), 137–154 (2005)
6. L'Homme, M.C., Bae, H.S.: A Methodology for Developing Multilingual Resources for Terminology. In: Language Resources and Evaluation Conference (LREC 2006), pp. 22–27 (2006)
7. L'Homme, M.C., Drouin, P.: Corpus de Informática en Español. Groupe Éclectik, Université de Montréal, `http://www.olst.umontreal.ca/`
8. Loper, E., Bird, S.: NLTK: The natural language toolkit. In: ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics, pp. 62–69 (2002)
9. Medina, A., Sierra, G., Garduño, G., Méndez, C., Saldaña, R.: CLI. An Open Linguistic Corpus for Enineering. In: IX Congreso Iberoamericano de Inteligencia Artificial (IBERAMIA), pp. 203–208 (2004)
10. Mima, H., Ananiadou, S.: An application and evaluation of the C/NC-value approach for the automatic term recognition of multi-word units in Japanese. International Journal on Terminology 6(2), 175–194 (2001)
11. Sager, J.C.: Commentary by Prof. Juan Carlos Sager. In: Rondeau, G. (ed.) Actes Table Ronde sur les Problémes du Découpage du Terms, pp. 39–74. AILA-Comterm, Office de la Langue Française, Montréal (1978)
12. Schmid, H.: Improvements in Part-of-Speech Tagging with an Application to German. In: ACL SIGDAT-Workshop (1995)