Tokenization and Proper Noun Recognition for Information Retrieval

Fco. Mario Barcala, Jesús Vilares, Miguel A. Alonso, Jorge Graña, Manuel Vilares
Departamento de Computación, Universidade da Coruña
Campus de Elviña s/n, 15071 La Coruña, Spain
{barcala,jvilares}@mail2.udc.es, {alonso,grana,vilares}@udc.es

Abstract

In this paper we consider a set of natural language processing techniques that can be used to analyze large amounts of texts, focusing on the advanced tokenizer which accounts for a number of complex linguistic phenomena, as well as for pre-tagging tasks such as proper noun recognition. We also show the results of several experiments performed in order to study the impact of the strategy chosen for the recognition of proper nouns.

1 Introduction

In recent years, there has been a considerable amount of interest in using Natural Language Processing (NLP) in Information Retrieval (IR) research, with specific implementations varying from the word-level morphological analysis to syntactic parsing to conceptual-level semantic analysis.

In this paper we consider the employment of a set of NLP techniques adequate for dealing with large amounts of texts. We propose the following sequence of finite-state based processes, each of them corresponding to the recognition of intuitive linguistic elements that reflect important universals about language:

- A *preprocessor* that identifies individual words, proper nouns and idioms forming each sentence.
- A *tagger* that assign a syntactic category to each word, in order to identify those words carrying the semantics of the sentence: nouns, adjectives and verbs.
- A *morphological families generator* that groups related words belonging to different categories (e.g. the noun corresponding to the action of a verb).
- A *shallow parser* that extract the basic syntactic structures relating words within a sentence, such as the noun-modifier, subject-verb or verb-object relations.

This paper is focused on the description of the preprocessor module, making emphasis in proper noun recognition tasks. Albeit our scheme is oriented towards the indexing of Spanish texts, it is also a proposal of a general architecture that can be applied to other languages with very slight modifications. To facilitate comprehension, English examples are used when possible.

2 The preprocessor

Most current systems assume that input texts are already tokenized, i.e. correctly segmented in *tokens* or high level information units that identify every individual component of the texts. This working hypothesis is not realistic due to the heterogeneous nature of the application texts and their sources. For this reason, we have developed a preprocessor module, an advanced tokenizer which accounts for a number of complex linguistic phenomena, as well as for pretagging tasks. The architecture of the preprocessor is shown in Fig.1, consisting of the following submodules.

Filter. This submodule performs the conversion from source format (e.g. HTML or XML) to plain text, and compacts delimiters (e.g. it removes multiple blanks or blanks at beginning of sentences).

Tokenizer. The main function of this submodule is to identify and separate the tokens present in the text, in such a way that every individual word as well as every punctuation mark will be a different token. The submodule considers abbreviations, acronyms, numbers with decimals, or dates in numerical format, in order not to separate the dot, the comma or the slash (respectively) from the preceding and/or following elements. For this purpose, it uses two dictionaries (one of abbreviations and another one of acronyms), and a small set of rules to detect numbers and dates.

Phrase segmentator. This submodule identifies sentences [7, 8]. The general rule consists of separating a sen-

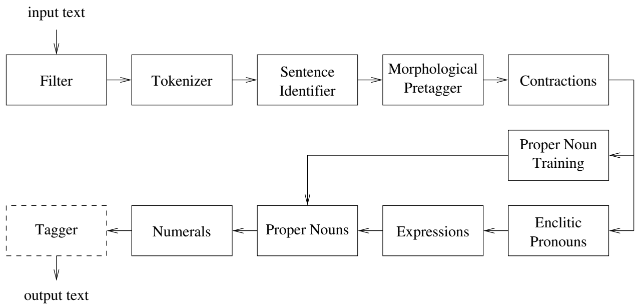

Figure 1. General architecture of the preprocessor.

tence when there is a dot followed by a capital letter. However, it must be taken into account certain abbreviations to avoid marking the end of a sentence at their dots. For instance, this is the case of `Mr. González`. The submodule also considers acronyms so as not to separate their individual capital letters.

Morphological pretagger. The function of this submodule is to tag elements whose tag can be deduced from the morphology of the word, and there is no more reliable way to do it. In this step, for instance, numbers and dates are identified.

Contractions. This submodule splits a contraction into their different tokens. At the same time, it assigns a tag to every one of them, by using external information on how contractions are decomposed. The submodule can work over other languages just by changing this information. For instance, the Spanish contraction `del` (*of the*) is decomposed into the preposition `de` (*of*) and the article `el` (*the*).

Enclitic pronouns. This submodule analyses the enclitic pronouns that appear in verbal forms. The objective is to separate the verb from its pronouns and tag every one of them correctly. In order to perform this function, this submodule uses the following:

- A dictionary with as many verbal forms as possible.
- A dictionary containing the greatest possible number of verbal stems capable of presenting enclitic pronouns.
- A list with all the valid combinations of enclitic pronouns.
- A list with the whole set of enclitic pronouns, together with their tags and lemmas.

For instance, the Spanish word `comerlo` (*to eat it*) is decomposed in `comer` (which is the infinitive *to eat*) and `lo` (which is the pronoun *it*).

Expressions. This submodule joins together the different tokens that make up an expression [2]. It uses two dictionaries: the first one with the expressions that are uniquely expressions, e.g. `a pesar de` (*in spite of*), and the second one with those that may be expressions or not, e.g. `sin embargo` (*however* or *without seizure*). In this case, the preprocessor simply generates all the possible segmentations, and then the tagger selects one of those alternatives later.

Numerals. This submodule joins together several numerals in order to build a compound numeral and so obtain only one token. Unlike the case of expressions, the tag assigned by the preprocessor here is definitive.

Proper noun trainer. Given a sample of the texts that are going to be indexed, this submodule learns a set of candidate proper nouns that are stored in the *trained dictionary*. This submodule identifies the words that begin with a capital letter and appear in non-ambiguous positions, i.e. in positions where if a word begins with a capital letter then it is a proper noun. For instance, words appearing after a dot are not considered, and words in the middle of the text are considered. These words are added to a dictionary which is used later by the Proper Noun Identifier submodule.

It also identify sequences of capitalized words connected by some valid connectives like the preposition *of* and definite articles. All possible segmentations of these sequences are considered. For example, for `High Council of Chambers of Commerce` the following proper nouns would be generated:

```
High&Council&of&Chambers&of&Commerce
High&Council
High&Council&of&Chambers
Council&of&Chambers&of&Commerce
Chambers&of&Commerce
```

where `&` is used to join the words that form the compound proper noun. Then, all these proper nouns are added to the trained dictionary of proper nouns.

Proper noun identifier. This submodule uses an *external dictionary* of proper nouns to which the *trained dictionary* extracted by the Proper Noun Trainer submodule can be added. With this resources, this phase of the preprocessor is able to detect proper nouns whether simple or compound, and either appearing in ambiguous positions or in non-ambiguous ones.

In the case of non-ambiguous positions, we check for the longest sequence of valid connectives and capitalized words present in the external dictionary, assigning the tag corresponding to the leading capitalized word. If this fails, we assign a proper noun tag with gender under-specified to the longest sequence.

In the case of ambiguous positions, we check for the longest sequence of valid connectives and capitalized words present in the external dictionary, assigning the corresponding tag. If this fails, we check for the longest sequence in the trained dictionary, labeling the sequence as a proper noun with gender under-specified. If this also fails, the sequence is not tagged.

As an example, if we find `Javier Pérez del Río` in ambiguous position, supposing the training phase have found `Pérez&del&Río` in a non-ambiguous position and that `Javier` is found in the external dictionary as a masculine proper noun, the whole sequence `Javier&Pérez&del&Río` is tagged as a masculine proper noun.[1]

3 The tagger

The output of the preprocessor is taken as input by the tagger. Almost any kind of tagger could be applied. In our system, we have used a second order Markov model for part-of-speech tagging. The elements of the model and the procedures to estimate its parameters are based on Brant's work [1], incorporating information from external dictionaries [4] which are implemented by means of numbered minimal acyclic finite-state automata [3].

4 Morphological families

A *morphological family* is the set of words obtained from the same morphological root through derivational mechanisms. We have considered the derivational morphemes, the allomorphic variants of morphemes and the phonological conditions they must satisfy, to generate the set of morphological families from a large lexicon of Spanish words [12]. The resulting morphological families can be used as a kind of advanced, linguistically motivated stemmer for Spanish.

5 The shallow parser

Given a stream of tagged words, the parser module, described in [11], tries to obtain the *head-modifier* pairs corresponding to the most relevant syntactic dependencies: *noun-modifier*, relating the head of a noun phrase with the head of a modifier; *subject-verb*, relating the head of the subject with the main verb of the clause; and *verb-complement*, relating the main verb of the clause with the head of a complement. The kernel of the grammar used by the parser is inferred from the basic trees corresponding to noun phrases and their syntactic and morpho-syntactic variants [5].

6 Evaluation

The lack of a standard evaluation corpus has been a great handicap for the development of IR research in Spanish.[2] This situation is changing due to the incorporation in CLEF-2001[3] of a Spanish corpus (composed of news provided by a Spanish news agency) which is expected to become a standard. The techniques proposed in this paper have been integrated very recently, therefore, we could not participate in CLEF-2001 edition, but we are prepared to join competition in 2002. Due to the unavailability of the CLEF corpus, we have chosen to test our techniques over the corpus used in [13], formed by 21,899 news articles (national, international, economy, culture,...) with an average length of 447 words. The total size of the corpus is about 60 MB of text. We have considered a set of 14 natural language queries with an average length of 7.85 words per query, 4.36 of which were content words.

The techniques proposed in this article are independent of the indexing engine we choose to use. This is because we first conflate the document to obtain its index terms; then, the engine receives the conflated version of the document as input. So, any standard text indexing engine may be employed, which is a great advantage. Nevertheless, each engine will behave according to its own characteristics (indexing model, ranking algorithm, etc.). The results we show here have been obtained using SMART.

We have compared the results obtained by four different indexing methods: stemmed text after eliminating stopwords (*stm*), lemmatized text (*lem*), text conflated by means of morphological families (*fam*) and syntactic dependency pairs (*sdp*).

In Fig. 2 we show the results obtained for these four techniques with the preprocessor module recognizing compound proper nouns. We can observe that *lem* improves the results of *stm* in precision, while *fam* improves the results of *stm* both in precision and recall. On the other hand, *sdp* gets worse results in recall though its precision is slightly higher

[1] In Spanish, `Javier` is a traditional first name, `Pérez` is a traditional family name, `del` is the resulting of contracting the preposition `de` (*of*) and the definite article `el` (*the*), and `Río` is the common noun *river*. The use of common nouns as part of a family name (in this case `Pérez del Río`) is a typical phenomenon in Spanish.

[2] The test collection used in the Spanish track of TREC-4 (1995) and TREC-5 (1996), formed by news articles written in Mexican-Spanish, is no longer freely available.

[3] `http://www.clef-campaign.org`

	stm	*lem*	*fam*	*sdp*
Average precision	0.2179	0.2214	0.2321	0.2656
Average recall	0.6335	0.6179	0.6459	0.4625

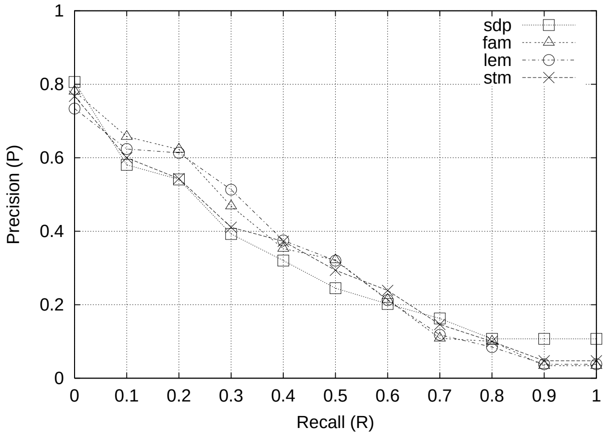

Figure 2. Results with compound proper nouns.

than for the other techniques, showing an improvement of 21.9% with respect to *stm*.

We have investigated the effects of the preprocessor module in the results, concluding the tasks performed by the Proper Noun Identifier submodule have a great impact in the performance of *sdp*. The main problems are:

1. The presence of different name forms to designate the same entity. For example *George Bush*, *G. W. Bush* and *Bush* refers to the same person but the submodule generates different terms for them.

2. Entities that are sometimes considered as common nouns, written in low letters, and sometimes as proper names, written capitalized. For example we can found *Secretaría General* and *secretaría general* (general secretaryship). The effect on indexing is important: when indexed as common noun, all words involved are conflated via morphological families, which makes possible to match nouns formed by derivatives, as in the case of *secretario general* (general secretary).

3. Entities with complex proper names, for example *Ministerio de Educación y Cultura* (Ministry of Education and Culture). If it is indexed as a single term, it can not match queries referring to the Minister of Education or to the Minister of Culture, as often occurs. In addition, such complex nouns form structured noun phrases by themselves, but any dependency structure can be extracted when they are considered as proper nouns.

	stm	*lem*	*fam*	*sdp*
Average precision	0.2179	0.2214	0.2268	0.2931
Average recall	0.6335	0.6211	0.6397	0.5179

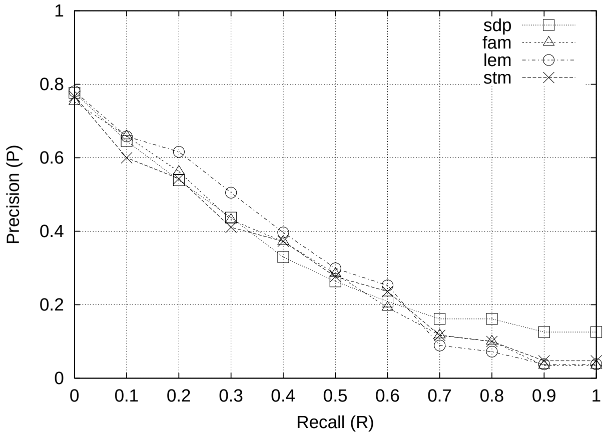

Figure 3. Results with simple proper nouns.

In order to solve these problems, we have decided to change the way proper nouns are managed: instead of tagging a sequence of capitalized words as a compound proper noun, each individual word is tagged as a simple proper noun. Accordingly, the grammar used by the parser is modified to take into account that several consecutive proper nouns can appear in the text. This is necessary to ensure the right dependencies are extracted.

For each capitalized word W we apply the following algorithm, both in the Proper Noun Trainer and in the Proper Noun Identifier submodules:

if W appears in the dictionaries of proper nouns
then W is tagged as a proper noun
else if W appears in the lexicon with label T
    then W is tagged as a T
    else if W is in a non-ambiguous position
        then W is tagged as an proper noun
        else W is tagged as an unknown word

The Proper Noun Trainer can only consult the external dictionary of proper nouns, while the Proper Noun Identifier can consult the external and the trained dictionary.

Applying this algorithm, *George W. Bush* is tagged as a sequence of three proper nouns, *Secretaría General* is tagged as commonNoun-adjective and *Ministerio de Educación y Cultura* is tagged as commonNoun-preposition–commonNoun-coordination-commonNoun.

In Fig. 3 we show the results obtained considering the modifications in the preprocessor and parser modules. The results for *lem* and *fam* remains almost the same, but *sdp* shows an important increase in precision and recall. In fact, the precision of *sdp* shows an improvement of 34.5% with respect to *stm*.

7 Conclusions

We have described in detail a preprocessor module for the right segmentation of texts which accounts for a number of complex linguistic phenomena, including the recognition of proper nouns. This module is the first stage in a cascade of natural language modules that, jointly with a search engine, makes up an information retrieval system.

Some other authors have investigated the impact of name recognition in information retrieval systems. Pfeifer et al. study in [9] the effectiveness of several methods of single surname search methods.

Thompson and Dozier [10] discuss the effect on retrieval performance of indexing and searching personal names differently from non-name terms in the context of ranked retrieval. They do not indexed personal names found in texts. Instead, they identified those ones present in the query, measuring retrieval performance with name searching simulated by searching with a proximity operator: they must appear, in the indicated order, within two non-stop words of each other. In contrast with our automatic approach, the approaches shown in [9, 10] worked with manually marked personal names.

Kwak et al. present in [6] a corpus based learning method that can index diverse types of (common) compound nouns using rules automatically extracted from a large tagged corpus. We have not followed this approach due to the formation of proper names can be easily defined my means of few hand written rules that take into account sequences of capitalized words.

Acknowledgments

This research has been partially supported by Plan Nacional de Investigación Científica, Desarrollo e Innovación Tecnológica (TIC2000-0370-C02-01), Ministerio de Ciencia y Tecnología (HP2001-0044) and Xunta de Galicia (PGIDT01PXI10506PN).

References

[1] T. Brants. TnT - a statistical part-of-speech tagger. In *Proceedings of the Sixth Applied Natural Language Processing Conference (ANLP'2000)*, Seattle, 2000.

[2] J.-P. Chanod and P. Tapanainen. A non-deterministic tokeniser for finite-state parsing. In *Proceedings of the Workshop on Extended finite state models of language (ECAI'96)*, Budapest, Hungary, 1996.

[3] J. Graña, F. M. Barcala, and M. Alonso. Compilation methods of minimal acyclic automata for large dictionaries. In B. W. Watson and D. Wood, editors, *Proc. of the 6th Conference on Implementations and Applications of Automata (CIAA 2001)*, pages 116–129, Pretoria, South Africa, July 2001.

[4] J. Graña, J.-C. Chappelier, and M. Vilares. Integrating external dictionaries into stochastic part-of-speech taggers. In *Proceedings of the Euroconference Recent Advances in Natural Language Processing (RANLP 2001)*, pages 122–128, Tzigov Chark, Bulgaria, 2001.

[5] C. Jacquemin and E. Tzoukermann. NLP for term variant extraction: synergy between morphology, lexicon and syntax. In T. Strzalkowski, editor, *Natural Language Information Retrieval*, volume 7 of *Text, Speech and Language Technology*, pages 25–74. Kluwer Academic Publishers, Dordrecht/Boston/London, 1999.

[6] B.-K. Kwak, J.-H. Kim, G. Lee, and J. Y. Seo. Corpus-based learning of compound noun indexing. In J. Klavans and J. Gonzalo, editors, *Proc. of the ACL'2000 workshop on Recent Advances in Natural Language Processing and Information Retrieval*, Hong Kong, Oct. 2000.

[7] A. Mikheev. Document centered approach to text normalization. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'2000)*, pages 136–143, Athens, Greece, 2000.

[8] A. Mikheev. Tagging sentence boundaries. In *Proceedings of the First Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL'2000)*, pages 264–271, Seattle, 2000.

[9] U. Pfeifer, T. Poersch, and N. Fuhr. Retrieval effectiveness of proper name search methods. *Information Processing and Management*, 32(6):667–679, 1996.

[10] P. Thompson and C. C. Dozier. Name recognition and retrieval performance. In T. Strzalkowski, editor, *Natural Language Information Retrieval*, volume 7 of *Text, Speech and Language Technology*, pages 261–272. Kluwer Academic Publishers, Dordrecht/Boston/London, 1999.

[11] J. Vilares, F. Barcala, and M. A. Alonso. Using syntactic dependency-pairs conflation to improve retrieval performance in Spanish. In A. Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, volume 2276 of *Lecture Notes in Computer Science,*, pages 381–390. Springer-Verlag, Berlin-Heidelberg-New York, 2002.

[12] J. Vilares, D. Cabrero, and M. A. Alonso. Applying productive derivational morphology to term indexing of Spanish texts. In A. Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, volume 2004 of *Lecture Notes in Computer Science*, pages 336–348. Springer-Verlag, Berlin-Heidelberg-New York, 2001.

[13] J. Vilares, M. Vilares, and M. A. Alonso. Towards the development of heuristics for automatic query expansion. In H. C. Mayr, J. Lazansky, G. Quirchmayr, and P. Vogel, editors, *Database and Expert Systems Applications*, volume 2113 of *Lecture Notes in Computer Science*, pages 887–896. Springer-Verlag, Berlin-Heidelberg-New York, 2001.