The PoS Tagger of Turin University

Leonardo Lesmo

Dipartimento di Informatica, Università di Torino
Corso Svizzera 185, I-10149 Torino, Italy
lesmo@di.unito.it

Abstract. This paper describes the Part of Speech Tagger that is part of the parsing system of Turin University. The PoS Tagger is based on a set of disambiguation rules developed manually. The rules are grouped into *ambiguity sets*, each of which is associated with a set of PoS. Some intra-category rules are defined to assign features to words. The paper includes a brief description of the morphological analyzer, and then focuses on the PoS Tagger organization.

Keywords: Rule-based systems, Part of Speech Tagging

1 Introduction

The PoS Tagger that is presented here is one of the components of a parsing system that produces a dependency tree for each sentence composing an input text. The organization of the parser and its results in the Evalita 2009 Parsing Task are described in [1]. The system includes a number of modules, some of which will not be presented here, since this description will be focused on the step that takes as input a list of possibly ambiguous items that contain various kinds of morphological information and chooses the best lexical hypothesis. The morphological analyzer will be briefly overviewed here, while all parsing steps that follow the application of the PoS Tagger will be skipped (see [1]).

A scheme of the whole system is reported in fig.1. All modules not included in the dashed box are described in some detail in [1].

Since this paper acts more as a system description than as a full scientific paper, we will not attempt here to survey the literature on PoS Tagging, which would require more space and an introduction to statistical approaches; the interested reader is addressed to the [2], and to the references cited therein.

The paper is organized as follows. In the next section, we briefly present the morphological analyzer that provides the input to the PoS Tagger. In the third section, we describe the organization of the tagger. In the fourth one, we discuss the performances of the tagger in the Evalita 2009 contest. A short Conclusion section closes the paper.

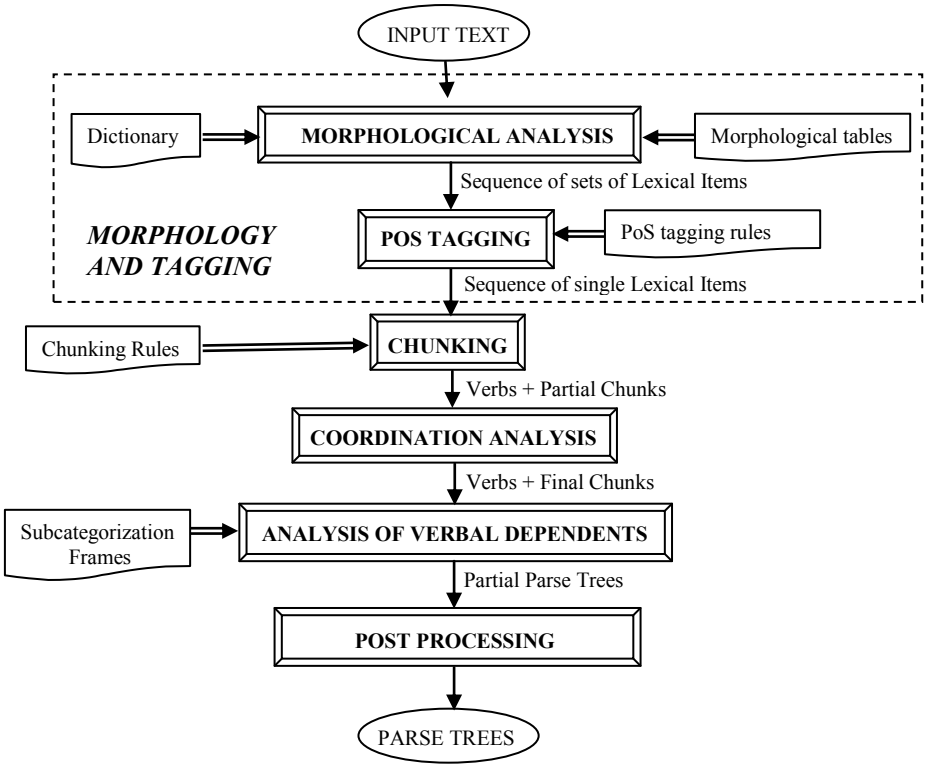

Fig. 1. Architecture of the parser (including components not described in this paper)

2 Morphological analysis

2.1 Tokenization

The input text consists in a sequence of characters (encoded either in ISO-Latin-1 or UTF-8) and undergoes a first step of tokenization. The tokenizer splits the text in words and sentences, and returns a sequence of (possibly ambiguous) token items. The procedure is based on an automaton that scans the sequence of characters and emits hypotheses about standard words, numbers, punctuation marks, abbreviations, and so on. Each token item is composed of a sequence of characters and a *Token Type*. The existing token types are: *GW* (general word), *NOMEP* (proper names), *SIGLA* (abbreviations, codes, initials), *NUMBER* (numbers, in digits), *DATE* (dates in a standard forms, as 18/01/2009), *SEGNOINTER* (punctuation marks), *SPECIAL* (other symbols, e.g. @), *PARAGRAPH-NUMB* (chapter or paragraph numbers, in dot notation, as 3.1.3.), *YEAR* (just for forms as '03, standing for 2003).

Possible ambiguities may arise for strings as "Barbara.". In this case, the output is the one shown in Table 1. The first token item includes two components: the first for the GW "barbara" (barbarianf,sing), and the second for the period (SEGNOINTER).

The second item is similar, but the first component concerns the proper name "Barbara". The third item refers to the hypothesis that "Barbara." is a single code,

including the period (as "Mr."). The final output is a sequence of items (see Table 1), which are then sent to the morphological analyzer for the access to the dictionaries[1].

Table 1. Tokenizer output for the string "Barbara."

((((98 97 114 98 97 114 97) GW)	** *first tokenizer item*
((46) SEGNOINTER))	
(((66 97 114 98 97 114 97) NOMEP)	** *second tokenizer item*
((46) SEGNOINTER))	
(((66 97 114 98 97 114 97 46) SIGLA)))	** *third tokenizer item*

2.2 Dictionaries

Each tokenizer item is used for accessing the dictionaries that constitute the lexical knowledge sources used by the parser. The base dictionary concerns general words and is indexed on word roots (stems). It includes 23.850 roots that correspond to 24.280 "normalized forms". The number of lemmata is slightly greater than that, since a single "normalized form" can correspond to more than one lemma (as, for instance, "corso", which is both a noun and an adjective, i.e. "avenue" and "Corsican"). For instance, the dictionary entry associated with the stem "cors" is the following:

(cors ((corso cat noun classe (2))
(corsa cat noun classe (1))
(corso cat adj classe (1))
(correre cat verb classe (8 (c (1 3 6) i))))

There are four entries: the first one for the noun 'corso' (avenue), the second one for the noun 'corsa' (run), the third one for the adjective 'corso' (Corsican), the last one for the verb 'correre' (to run). The *classe* information refers to the possible affixes that can be attached to the root to produce the various forms. For instance, the second nominal class is associated with *-o* (masculine singular) and *-i* (masculine plural). For verbs, it is possible to specify the tenses to which the stem applies. In the example, it is said that this stem is used for the first person singular (1), for the third person singular (3) and for the third person plural (6) of the past (c) and for the past participle (i). The affixes are the ones associated with class 8 (e.g. *cors-i, cors-e, cors-ero* for the past). The information about the affix classes is stored in a separate file, where 21 adjectival classes, 32 nominal classes and 12 verbal classes are defined.

Apart from the base morphological dictionary, other dictionaries are in use:

1. Italian proper names (e.g. Giuseppe, Parigi, …): 248 entries
2. Language-independent proper names (e.g. London, Joyce, …): 384 entries
3. Multi-word expressions (e.g. 'a prima vista' – at first sight): 322 entries
4. Italian multi-word proper names (e.g. 'Costa d'Avorio' – Ivory Coast): 56 entries
5. Language-independent multi-word proper names (e.g. Los Angeles): 130 entries

[1] A second automaton takes care of the analysis of numbers written in letters, as "threehundredfortyfive". The tokenizer automaton includes 89 nodes, while the numbers automaton includes 88 nodes.

3 Part of Speech Tagging

After the morphological analysis, we know, for each input item, the set of possible interpretations. For instance, if the input word is *corso*, we have:

(corso ((corso cat noun gender m number sing)
(corso cat adj gender m number sing)
(correre cat verb mood participle tense past gender m number sing)))

Where the various features are obtained on the basis of the class and of the suffix *+o* (*cors+o*). The entry shown above licenses three possible PoS: *noun*, *adj*, *verb*. They form the *ambiguity set* of the entry. For each ambiguity set, a group a rules and a default category are defined (for instance, the ambiguity set of *adj+noun+verb* includes 38 rules and the default is *noun*).

The sequence of word interpretations is scanned from left to right. For each word, the ambiguity set is determined and the corresponding packet is activated.
Each rule has the following form:

if condition *then* category *CF* certainty-factor

Where the 'condition' is described below, 'category' is one of the categories to which the packet is associated and 'certainty-factor' is one of *C* (Certain), *A* (Almost certain), and *U* (Uncertain). Since the first rule that fires is the one used to choose the category, the certainty factor provides a very rough way to define the order of application of the rules. For rules having the same certainty factor, the order in which they are listed (manually) defines the order of application. An example of rule is:

if (AND (nextcat-agr noun adj)
(not (currword-typ &adj-pref-noun))
(prevcat verb))
then noun
CF U

This example enables us to present the main features of conditions for rule application. The usual logical operators *AND*, *OR*, *NOT* can be used to build complex conditions on the basis of elementary predicates. The latter can involve:

a. the inspection of the features of the word under analysis
b. the inspection of the categories and features of the two preceding and following words (possibly enforcing some agreement check). This may require the access to a dedicated KB of *special words*[2]
c. in very limited cases, the search for an item in the whole sentence, as, for instance, a final question mark.

Before providing some examples of elementary predicates, it is worth having a look at the KB of special words. It includes 198 groups of words, whose role is to act as syntactic sub-sub-categories, some of which have a semantic flavor. An example of the latter is the category *&title*, that includes the words *ministro, presidente, signore, commissione, dottore, commendatore*. A more syntactically oriented class is *&conj-governing-subjunctive*, including the words *purché* and *affinché*, which are conjunctions governing sentences whose verb should be in the subjunctive mood.

[2] Actually, this KB is also used in various modules of the parser

With respect to elementary predicates, since the PoS Tagger proceeds strictly left-to-right, all checks on preceding words refer to items already disambiguated, while the tests on the following words refer to possibly ambiguous items.
Examples of the first type of predicates are:

- (*currmood args*), which checks if the mood of the current word (a verb) is among the ones in *args*.
- *(currtype args)*, which checks if the type (subcategory) of the current word is among the ones in *args*.

Examples of the second type are:

- (*nextcat-agr next-cat current-cat*), which checks if among the possible interpretation of the next word there is one item of category *next-cat*, and that item agrees (usually in gender and number) with the interpretation of the current word of category *current-cat*.
- *(prev2word-typ subsubcat)*, which checks if the (already disambiguated) word that occurs two words before the current one belongs to the group *subsubcat*.

Examples of the third type are[3]:

- *(interr-sent)*, which checks if the sentence under analysis includes an interrogative marker (that, for Italian and English is a final question mark).
- *(beforecat category)* which checks if in a larger window before the current word (currently defined as 10 words) there occurs a word of category *category*.

Before closing this section, we mention the existence of intra-category rules. They have the same format seen above, and are applied in case more than one interpretation is associated with a given category. An example of packet of this type is *verb-main-aux*, that contain rules for disambiguating between the auxiliary and non-auxiliary interpretations of the verbs *essere* (to be)*, avere* (to have), etc.

4 The PoS Tagger at Evalita 2009

Although the PoS Tagger, in the context of the whole parsing system, usually works rather well, its rank in the Evalita 2009 Tagging Task was rather low (6th among 8 participants). In this section, we try to account for this result, by analyzing in some details the errors made by the tagger. The total number of errors is 199 on a test set of 4919 words, i.e. 4.05%. The resources used by the tagger in Evalita are the same that it uses during normal operations, with the exception of the dictionary of multi-word expressions. This was done in accordance with the data in the development set, but had some negative consequences, as we will see below.

Unknown words. The first point to notice is that currently the tagger does not handle unknown words. A simple way to face the problem would be to assign to all unknown words an ambiguity set consisting in all (or most) open classes (i.e. *verb, noun, adj, adv*) and make the tagger apply the standard rules of this class. However, this approach requires some tests and we wanted to avoid architectural changes made only in the specific context of Evalita. So, unknown words undergo just a post-processing step, where words ending with a consonant are supposed to be foreign words. There were 48 errors due to unknown words plus 6 errors concerning foreign words. Eight

[3] Currently, there are only 18 occurrences of predicates of this type in the whole set of rules

of these errors are due to the non-use of multi-words. For instance, there are 5 occurrences of "a seconda di/che" (according as), where the required adverbial interpretation of "seconda" does not exist out of this context, so that it was incorrectly classified as an adjective (second). With respect to foreign words, the tag FW was taken as wrong for the word "sir" (2 occurrences).

Participles vs. adjectives. 10 errors are due to the wrong tagging of participles as adjectives, or viceversa. Although the criterium adopted by the organizers (the absence of a depending phrase implies adjective, otherwise verb) is reasonable, there are some cases that are at least dubious. An example is "sono basate sulla rilevazione" (are based on the survey), where "basate" seems to be an adjective, and not a verb.

Gender. 6 errors are due to the use of the 'n' (neuter) feature of nouns. For instance "via" (road or start) is tagged as 'n', because the two senses of the word are one feminine and one masculine. However, the same does not happen for "varietà" (variety or variety show), which is tagged in the Gold dataset as feminine.

Gold errors. There are 12 errors in the Gold dataset, where we also include disagreements with the Development set. A noticeable case refers to 4 occurrences of the 'single quote' mark ('), which is consistently tagged in the development set as FB, whereas they are tagged as FF in Gold.

Of course, most of the difficulties listed above are common to all participants to the Tagging task, so nothing changes about the rank of the Turin tagger. This only helps explaining the difference between the standard performances of the tagger (which are consistently above 96.5%) and the ones obtained for Evalita (95.95%)

5 Conclusions

We do not have information about the organization of the other PoS Tagger participating in Evalita 2009, but it is reasonable to assume that they are based on statistical techniques. The overall results seem to confirm that rule-based and statistics-based system achieve comparable level of performances. One of the point that must be stressed is that Evalita 2009 was particularly useful for us, since it enabled us to identify the main shortcomings of the system. Apart from the mentioned troubles on unknown words, many residual errors are real tagging errors that were not revealed by the tests on the Turin corpora. An example concerns the occurrence of quotation marks, that very often, but not always, should be disregarded to obtain the correct tag. We look forward to the next Evalita contest to get even better results.

References

1. Lesmo, L.: The Turin University Parser at Evalita 2009. In: Proceedings of EVALITA 2009 (2009)
2. Voutilainen, A.: Part-of-Speech Tagging. In: Jurafsky, D., Martin J. H.: Speech and Language Processing (2nd Ed). Prentice Hall, pp.219-232 (2008)