Tokenization of Portuguese: resolving the hard cases

António Branco
João Silva

DI-FCUL TR–2003–4

March 2003

Departamento de Informática
Faculdade de Ciências da Universidade de Lisboa
Campo Grande, 1749-016 Lisboa
Portugal

Technical reports are available at http://www.di.fc.ul.pt/tech-reports. The files are stored in PDF, with the report number as filename. Alternatively, reports are available by post from the above address.

Tokenization of Portuguese: resolving the hard cases

António Branco and João Ricardo Silva

Department of Informatics, University of Lisbon

This research note addresses the issue of ambiguous strings, strings of non-whitespace characters whose tokenization, depending of the specific occurrence, yields one or more than one token. This sort of strings, typically coinciding with orthographically contracted forms, is shown to raise the problem of undesired circularity between tokenization and tagging, under the standard view that tokenization takes place before tagging. The critical importance of this apparently minor, low-level issue results from the fact that these strings correspond mostly to functional words, that they are quite frequent, covering over 2% of a corpus, and that their careless treatment would introduce unrecoverable degradation of performance at a very early stage of language processing and that this degradation would trigger further and wider loss of accuracy in all subsequent processing stages.
We argue for a resolution of this circularity on the basis of a new, two-level approach to tokenization. This approach is shown to be used also to improve the problem of sentence chunking at periods that are ambivalent between marking the end of an abbreviation and the end of a sentence.

1 Introduction

Part-of-speech tagging is a task by means of which each linguistically relevant token is associated with the appropriate morphosyntactic category (e.g. Noun, Verb, Conjunction, etc.) given its specific occurrence. Tagging, and a fortiori every subsequent processing task in Natural Language Processing (NLP), presupposes a preliminary phase of tokenization after which the input sequence is provided with its individual tokens explicitly identified and isolated from each other.
The tokenization task is thus a low-level procedure in the spectrum of the NLP levels of processing, typically proceeding by simple character

recognition and manipulation operations. It can be envisaged as having to handle two major cases:
(i) a trivial one, where there is the confirmation that the whitespaces between words delimit tokens: `um exemplo` → `|um|exemplo|`;
(ii) a less trivial one, where more than one token is to be identified in a string not containing a whitespace, a case that can be further detailed into:
(ii.i) a mere separation of two adjacent tokens:

`um,` → `|um|,|`, `viu-a` → `|viu|-a|`;

(ii.ii) the identification and reconstruction of tokens out of a single sequence of characters:

`à` → `|a|a|`, `pelos` → `|por|os|`, `lhas` → `|lhe|as|`.

When applying this standard scheme to Portuguese, one finds issues to be tackled that are far from being as simple as it might be expected at first glance: There are cases of puzzling circularity where items should have been previously tagged for them to be correctly tokenized, but for them to be tagged, as made explicit above, they should have already been tokenized.
The hard cases for tokenization of Portuguese texts involve ambiguous strings: Strings of non-whitespace characters, typically coinciding with orthographical contractions of word forms, that depending on the specific occurrence, are to be considered as consisting of one or more than one token.
Though these strings types are little more than a dozen, the critical importance of their accurate tokenization results from the fact that: (i) their tokens are quite frequent - covering over 2% of a corpus - and that (ii) they correspond mostly to functional words, thus implying that their careless treatment would introduce unrecoverable degradation of performance at a very early stage of language processing and that this degradation would trigger further and wider loss of accuracy in all subsequent processing stages.
This research note addresses these hard cases in the tokenization of Portuguese texts. It presents a strategy for resolving the undesired tagging-tokenization circularity on the basis of a new, two-level approach to tokenization. This strategy is shown also to improve the problem of sentence chunking at periods that are ambivalent between marking the end of an abbreviation and the end of a sentence.
In Section 2, we identify and describe in detail the problematic cases of circularity between tagging and tokenization.

In Section 3, we present a solution to dissolve this circularity that is based on two levels of tokenization, one preceding and another following the tagging phase, and present an evaluation of this solution in Section 4.
This result is shown, in Section 5, to have interesting generality, beyond Portuguese tokenization, as it can be extended to help resolve other problems as well, namely problems concerning sentence chunking in the context of ambivalent periods in abbreviations, an issue that has to be dealt with for other natural languages.
Finally, in Section 6, we present our concluding remarks.

2 The tokenization-tagging circularity

As a starting point, let us take orthographic contractions in the more intensively studied natural language. In spite of the fact that in English, contractions do not exhibit a large diversity, the few existing cases are illustrative examples of the need to identify different tokens out of a single sequence characters not including whitespaces. For instance, in the Penn Treebank, the sequences `I′ m` and `won′ t` are tokenized as `|I|′ m|` and `|wo|n′ t|`, respectively [2].
If for the sake of facilitating subsequent uniform syntactic processing of every verb form, orthographic normalization is sought, then reconstruction of correct orthographic form of the identified tokens is also an issue here. Besides identification, reconstruction is needed so that the sequences `I′ m` and `won′ t` are handed over to tagging as `|I|am|` and `|will|not|`, respectively.
In Portuguese orthography, there are several instances of orthographic contractions. Most of such cases concern the contraction of a Preposition with the subsequent word.
The Prepositions *por* and *para* may contract with Definite Articles:

pelo (*por o*), *p'lo* (*por o*), *pr'à* (*para a*)

The Preposition *com*, in turn, has a special contracted form with Personal Pronouns in accusative declination:

c*ontigo* (*com ti*)

Other Prepositions may contract with items from a wider range of categories. That is the case of *de* and *em*, which contract with Definite Articles and also with Indefinite Articles, Personal and Demonstrative Pronouns:

> *do* (*de o*), *dum* (*de um*), *dele* (*de ele*), *disto* (*de isto*), *disso* (*de isso*), *daquilo* (*de aquilo*)
> *no* (*em o*), *num* (*em um*), *nele* (*em ele*), *nisto* (*em isto*), *nisso* (*em isso*), *naquilo* (*em aquilo*)

Besides Prepositions, also Clitics either in proclisis or not, may be contracted with other clitics:

> *lho* (*lhe o*)

In this connection, the crucial point to note is that there are strings of characters that are ambiguous between a contracted and a non-contracted form. For instance, the string `pelo` is ambiguous between the first person singular of "Presente Indicativo" of Verb *pelar* and the contraction of the Preposition *por* and the Definite Article *o*.
The list of character strings in such circumstances contains at least a dozen of items. Like the examples above, there are strings that are ambiguous between a contraction of Preposition+Article, Preposition+Demonstrative Pronoun or Preposition+Personal Pronoun, and a Verb:

`pelo` → `|por|o|` or `|pelo|` (*pelar*, 1st person singular, "Presente Indicativo")
`pelas` → `|por|as|` or `|pelas|` (*pelar*, 2nd pers. sing., "Pres. Ind.")
`pela` → `|por|a|` or `|pela|` (*pelar*, 3rd pers. sing., "Pres. Ind.")
`deste` → `|de|este|` or `|deste|` (*dar*, 2nd pers. sing., "Pretérito Perfeito")
`desse` → `|de|esse|` or `|desse|` (*dar*, 1st and 3rd pers. sg, "Pres. Conjuntivo")
`desses` → `|de|esses|` or `|desses|` (*dar*, 2nd pers. sing., "Pres. Conj.")
`consigo` → `|com|si|` or `|consigo|` (*conseguir*, 1st pers. sing., "Pres. Ind.")

There is an item that is ambiguous between a contraction Preposition+Personal Pronoun and a Noun:

`nele` → `|em|ele|` or `|nele|` (masculine, sing)

Some other strings are ambiguous between a contraction Preposition+Article and a Clitic:

`no` → `|em|o|` or `|no|` (3rd pers., masc., sing., pronominal)
`na` → `|em|a|` or `|na|` (3rd pers., feminine, sing., pronominal)
`nos` → `|em|os|` or `|nos|` (3rd pers., masc., plural, pronominal)
`nas` → `|em|as|` or `|nas|` (3rd pers., fem., plural, pronominal)

Finally, ambiguity between a contraction Clitic+Clitic and a Conjunction may also be found:

`mas` → `|me|as|` or `|mas|`

These items introduce circularity in the standard scheme of tokenization-followed-by-tagging: They should have been previously tagged for them to be correctly tokenized — but for them to be tagged, they should have already been tokenized.
For instance, to decide whether `mas` should be tokenized as `|mas|` or as `|me|as|` in a specific occurrence, it is necessary to know whether in that specific occurrence `mas` is a Conjunction or the contraction of two Clitics, respectively: This presupposes that the tagging process should have taken place, but for this process to have taken place, tokenization was expected to have been applied, and the circle is closed. The same problem is found for each one of the items listed above, which for the sake of simplicity, we will be referring to as ambiguous strings.
The low level of the task of tokenization with regards to the spectrum of the NLP levels should not bias one to disregard the importance of this problem. From a performance point of view of any NLP task, this is far from being a minor or merely curious issue. On the one hand, these ambiguous strings include potential tokens of functional categories, which are known to be of the utmost importance to constraint the syntactic environment where they occur and to help guiding the subsequent parsing process. On the other hand, these items have a non-negligible frequency, accounting for as much as around 2.1% of a large enough corpus, as the one we used in the experiment reported in the next Section.[1] A careless, low precision approach to

[1] The 257 125 token corpus we used was prepared from a corpus provided by the CLUL-Centro de Linguística da Universidade de Lisboa containing excerpts from newspapers, magazines,

tackle this issue, at such an earlier stage of processing, is very likely to imply a severe and unrecoverable damage in the overall performance of subsequent tasks, and in particular, in the immediately subsequent task of tagging.
The importance of an accurate handling of the ambiguous strings can be noticed in detail by taking into account their relative frequencies and the relative frequency of each of the two options for their tokenization in a corpus:[2]

Table 1 – Distribution of ambiguous strings

Strings	Freq.	One token	Two tokens
consigo	17	8	9
desse	33	6	27
desses	14	0	14
deste	85	6	79
mas	1015	1015	0
na	1314	2	1312
nas	222	2	220
nele	11	0	11
no	1450	14	1436
nos	431	127	304
pela	356	0	356
pelas	69	0	69
pelo	397	0	397
Total	5414	1180	4234
		21.80%	78.20%

3 The two-level approach

In the present section, we report on an experiment to resolve this tokenizing-tagging circularity. This experiment sought to test a two-

proceedings of meetings and fiction novels. We are grateful to Fernanda Nascimento and Amália Mendes for their kind help in this experiment.

[2] Data relative to the corpus identified in the previous footnote

level approach to tokenization, which for the sake of simplicity, we will be referring as the tok-tag-tok approach.
According to this approach, tagging is interpolated into the tokenization process, which has now two levels, one before and another after the tagger has been applied. The core idea is the following. In the first step (pre-tagging), every string is definitely tokenized, except those that are ambiguous, like the ones listed above, which receive a temporary tokenization as a single token. In the second step (tagging), the tagger is applied. And in the third step (post-tagging), these ambiguous strings are definitely tokenized and possibly expanded into several tokens. Pre-tagging tokenization resolves every string that is not ambiguous, while. post-tagging tokenization resolves token-ambiguous strings, which have then already been disambiguated by the interpolated step of tagging.
For this tok-tag-tok approach to work, it requires a few ingredients.

• First, the set of part of speech tags (tag set) used in tagging has to be expanded with a few new tags. These tags are formed simply by concatenating the tags of the different individual tokens that may come out of the tokenization of an ambiguous string. Given the ambiguous strings above and their possible categories, our tag set was expanded with:

PREPDA (contraction of Preposition and Definite Article),
PREPDEM (contraction of Preposition and Demonstrative Pronoun),
PREPPRS (contraction of Preposition and Personal Pronoun),
CLCL. (contraction of Clitic and Clitic).

It was expanded also with new tags for the individual items of multi-word expressions, such as LADV12, LPREP23, etc. For instance, the string `no` in the three element multi-word sequence `no entanto` will receive the tag LADV12, and the whole sequence will be tagged as `no`_LADV12 `entanto`_LADV3.

• Second, the corpus used for training the tagger and the standard corpus, used for evaluation, differ from each other with respect to the tokenization and tagging of ambiguous strings.
In the training corpus, the ambiguous strings are tokenized as a single token and receive one of two tags. For instance, `desse` is tokenized as

|desse| in every occurrence, and it is tagged either as desse_V or desse_PREPDEM: The tag assigned depends on the fact that, in the relevant occurrence, desse is the non contracted form - corresponding to the verb *dar* -, or the contracted form – corresponding to the preposition *de* and the demonstrative pronoun *esse*.
In the standard corpus, an ambiguous string is tokenized as one token if it is not a contracted form, or as two tokens, otherwise, together with the corresponding tags. For instance, in the standard corpus, desse appears as desse_V or as de_PREP esse_DEM, depending on the specific occurrence.

• Third, the tokenizer is used as the pre-tagging tokenizer, while another piece of code, the post-tagging tokenizer, has to be developed to process the output of the tagger. This second tool will look for items annotated by the tagger with the provisional tags PREPDA, PREPDEM, etc. and definitely tokenize and tag them as several tokens with the corresponding tags. For instance, upon an occurrence of no_PREPDA or no_LADV12, this post-tagging tokenizer will produce em_PREP o_DA or em_LADV1 o_LADV2, respectively.
Summing up, the overall tokenization and tagging process will proceed in the following way:

- a **pre-tagging tokenizer** definitely identifies every token except those related to ambiguous strings: These strings are provisionally identified as one token;
- the **tagger** assigns a composite or a simple tag to every ambiguous string depending on the fact that it is a contracted or a non contracted form, respectively: The tagger has been trained over a corpus where ambiguous strings are always tokenized as a single token and annotated with single or composite tags;
- a **post-tagging tokenizer** handles only ambiguous strings, breaking those that are tagged with a composite tag into two tokens and corresponding tags.

4 Evaluation

In our experiment, we implemented the tok-tag-tok approach using Ratnarparkhi's MXPOST system [3] to develop a tagger for Portuguese from an accurately hand tagged training corpus of around 250 000 tokens.[3]

As expected, the success of the experiment depended heavily on there existing or not a sparseness of the data bottleneck. Fortunately, the cases of ambiguous strings typically involve quite frequent tokens, which helped to find a suitable amount of training data and to greatly circumvent this problem.

The success of the experiment depended also on the performance of the tagger used. The training system we opted for offers a state of the art level of performance, having permitted to develop a tagger with 96.75% of success rate.

We evaluated this implementation of the tok-tag-tok approach by training the tagger over 90% of the corpus. A success rate of 99.44% was obtained with one run test over a held out evaluation corpus with the remainder 10%, obtained by extracting one out of each 10 consecutive sentences.

This is a very positive result. The baseline value for the success rate of the automation of the task of resolving ambiguous strings is 78.20%. This value is obtained by opting for tokenizing every ambiguous string as a sequence of two tokens (vd. Table 1, where 78.20% of all ambiguous strings occurred as contractions). The success rate obtained in our experiment, in turn, with a tagger with 96.75% accuracy, amounts to 99.44%, clearly a very significant improvement over the baseline value.

This result will be better if the tok-tag-tok approach is implemented with a tagger with higher accuracy. Naturally, in the limit, with a perfect tagger, the tok-tag-tok approach would allow to handle correctly every ambiguous string.

[3] Cf. description in footnote 1.

5 Abbreviations and sentence chunking

The tok-tag-tok rationale can be applied to other family of problems — occurring also in languages other than Portuguese — and typically conceptualized as being different in nature from the tokenization task, in particular as being of an even lower level than tokenization in the spectrum of the language processing stages.

The task of sentence chunking turns out not to be as straightforward as it might be assumed at first glance because one of the most frequent terminator characters, the period, is ambiguous [1]. It marks the end of sentences, as in the example `Este é um exemplo.`; it marks the end of an abbreviation as in the example `Estive seg. em Lisboa.`; and it marks both the end of a sentence and the end of an abbreviation, as in the example `Cheguei na seg. Lisboa estava calma.`.

The interesting point to note here is that these non-trivial cases for sentence chunking can be viewed, and analyzed, as particular instances of ambiguous strings. In the second example above, `seg.` is a single token and is expected to be tagged as `seg._WD`, while in the third example, `seg.` is the contraction of two tokens |`seg.`| and |`.`| which should yield `seg._WD` and `._PNT` when tagged. This clearly suggests that the tok-tag-tok approach might be successfully used to handle this family of non-trivial cases for chunking as well.

For this approach to be applied, the two occurrences above of `seg.` would receive a different annotation in a training corpus. In the first occurrence, `seg.` should be tagged as `seg._WD`, and in the second as `seg._WDPNT`. A tagger trained over such a corpus would be able to resolve the ambiguity of the period by assigning in one of the two tags. The post-tagging tokenizer that would subsequently act on the outcome of such tagger would leave untouched `seg._WD` but would separate `seg._WDPNT` into two tokens, `seg._WD` and `._PNT`.

There is here, however, a critical issue that does not arise in the processing of the ambiguous strings reported in the previous Section. The issue is that the conditioning of the tagger decision in subsequent steps is basically not affected when it has to tag, say `pelo` as `pelo_V` or as `pelo_PREPDA`. But this is not the case when the tagger has to tag `seg.`. If the period in `seg.` only marks the end of an abbreviation,

as in the sequence `seg. em`, the tagger decision for the subsequent string `em` will be conditioned by the context preceding that token, as desired. On the other hand, however, if the period also marks the end of a sentence, as in `seg. Lisboa` above, the tagger decision for the subsequent string `Lisboa` will also be conditioned by the context preceding it, contrarily to what is desired: The tagging decision for the first token of a sentence, `Lisboa` in the present case, should not be conditioned by the last tokens of the previous sentence.
A straightforward way out of this problem consists simply in enlarging the set of markers recognized as signaling the beginning of a sentence by the tagger. Typically, this set includes one or more consecutive symbols for new line. It should be extended to include also the tags *PNT.
The ambiguity of periods is an issue not restricted to Portuguese and applying the tok-tag-tok approach to abbreviations would help to resolve this sort of ambiguity in other languages as well. Given the typical sparseness of the data relevant for training a tagger for this kind of ability, we anticipate, however, that the success of this solution will be of practical use only if there is a very large training corpus, and in any case, that its success will be very dependent on its application being restricted to narrow language domains.

5 Concluding remarks

In this research note we addressed the issue of ambiguous strings, strings of non-whitespace characters whose tokenization, depending of the specific occurrence, yields one or more than one token. The critical importance of this apparently minor, low-level issue results from the fact that these strings typically correspond to functional words, that they are quite frequent, covering over 2% of a corpus, and that their careless treatment introduce unrecoverable degradation of performance at a very early stage of language processing, degradation that propagates and triggers further degradation at subsequent processing stages.
We showed that this type of strings raises the issue of undesired circularity between tokenization and tagging under the standard conception according to which tokenization takes place before tagging. We argued for a resolution of this circularity on the basis of a two-level

approach to tokenization. In the first, pre-tagging level, ambiguous strings are isolated as one token, and in the second, post-tagging level, each of these strings is either tokenized as two tokens or as one token, depending on the tag it was annotated with, indicating that it is a contracted or a non-contracted string of tokens, respectively.
We discussed also how the ambiguity of periods can be fruitfully conceptualized as a case of string ambiguity, and how the problem of sentence chunking at the occurrence of periods can be improved by using the two-level approach for tokenization, originally motivated by orthographically contracted forms.

References

1. Mikheev, Andrei: Periods, Capitalized Words, etc. Computational Linguistics 28(3). (2002) 289-318.

2. Mitchell, Marcus, Mary Marcinkiewicz, and Beatrice Santorini:Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics* 19(2) (1993) 313-330.

3. Ratnaparkhi, Adwait: A Maximum Entropy Model for Part-of-Speech Tagging, In Eric Brill and Kenneth Church (eds.), *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. ACL (1996) 133-142.