Contractions: Breaking the Tokenization-Tagging Circularity

António Horta Branco and João Ricardo Silva

University of Lisbon, Dept. of Informatics
{ahb,jsilva}@di.fc.ul.pt

Abstract. Ambiguous strings are strings of non-whitespace characters, typically coinciding with orthographic contractions of word forms, that depending on the specific occurrence, are to be considered as consisting of one or more than one token. This sort of strings is shown to raise the problem of undesired circularity between tokenization and tagging. This paper presents a strategy to resolve ambiguous strings and dissolve such circularity.

1 Tokenizing-Tagging Circularity

As a starting point, let us take orthographic contractions in the most widely studied natural language. In spite of the fact that in English, these items 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 and the strings `I'm` and `won't` should be tokenized as `|I|am|` and `|will|not|`, respectively.

In Portuguese orthography, there are several instances of orthographic contractions. Most relevant cases concern the contraction of a Preposition with the subsequent word. Some Prepositions contract with Articles and also Personal and Demonstrative Pronouns: *pelo (por* o), *nele (em el*e), *desse (de ess*e). Besides Prepositions, also some 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 some of these strings are ambiguous between a contracted and a non-contracted form (see list of ambiguous strings in Table 1). 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* (for the full list of ambiguities, see [1]).

N.J. Mamede et al. (Eds.): PROPOR 2003, LNAI 2721, pp. 167–170, 2003.

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 low level of the task of tokenization with regards to the spectrum of the NLP levels should not bias one to not recognize 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 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 shown 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
`no`	1450	14	1436
`nos`	431	127	304
`pela`	356	0	356
`pelas`	69	0	69
`pelo`	397	0	397
Total	5403	1180	4223
		21.84%	78.16%

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

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

2 Tokenization with Interpolated Tagging

In the present section, we report on an experiment to resolve this tokenizing-tagging circularity. This experiment sought to test a two-level approach to tokenization. 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 approach to work, it involves a few ingredients. First, the tag set 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, PREPPRS, and PREPDEM. 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 standard corpus and the training corpus for the tagger differ from each other with respect to the tokenization and tagging of ambiguous strings. In the training corpus, the ambiguous strings are provisionally tokenized as a single token and receive one of two tags. For instance, `desse` is tokenized as `|desse|` in every occurrence, but it is tagged either as `desse`_V or as `desse`_PREPDEM, depending on the fact that, in the specific occurrence, it is the non contracted form of the verb *dar* or the contracted form of the preposition *de* and the demonstrative pronoun *ess*e. 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 d*e*_PREP `esse`_DEM, depending on the specific occurrence.

Third, the usual tokenizer is transformed in the pre-tagging tokenizer, while another piece of code has to be developed to post-process the outcome of the tagger. This second tool will look for items annotated by the tagger with the provisional, but hopefully correct, tags PREPDA, PREPDEM, etc. and definitely tokenize and tag them as two tokens with the corresponding two 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.

In our experiment, we implemented this approach using Ratnarparkhi's MXPOST system [3] to develop a tagger for Portuguese from a training corpus of around 250 000 tokens.[3] As expected, the success of the experiment depends heavily on there

[3] Cf. description in footnote 1.

existing or not a sparseness of the data bottleneck and on the performance of the tagger used. 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 the possible sparseness problem. The training system we opted for, in turn, 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 our 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 approach described above is implemented with a tagger with higher accuracy. Naturally, in the limit, with a perfect tagger, it would allow to handle correctly every ambiguous string.

3 Further Results

The tokenization strategy described here – originally motivated by Portuguese orthographically contracted forms – can 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: The ambiguity of periods, occurring in several languages other than Portuguese, just need to be conceptualized as a case of string ambiguity. Check [1], for details on these further results.

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.