Swift Development of State-of-the-Art Taggers for Portuguese

António Branco and João Silva

Department of Informatics, University of Lisbon
Faculdade de Ciências, Campo Grande, 1749-016 Lisboa
{ahb,jsilva}@di.fc.ul.pt

The application of general-purpose machine learning techniques to natural language part-of-speech tagging has matured to a point where it is now quite rapid to develop new taggers. In the present paper, we report on solutions we adopted for the specific issues that arise when developing a new automatic tagger for Portuguese and are generic enough to be further reused to develop other new taggers for this language, possibly by using other training data.

Introduction

A basic linguistic generalization arises from the fact that some lexemes can replace each other without disrupting the grammaticality of whatever construction they may be part of, disregarding the fact that possible subcategorization and agreement constraints complied with by the replaced lexeme may not be observed by the replacing one. Lexemes under these circumstances are then said to have the same syntactic distribution and this generalization is signaled by grouping them in the same syntactic category, which is tantamount to assigning them the same part-of-speech (POS) tag, e.g. Noun, Adjective, Preposition, etc.

It happens that many lexemes belong to more than one such distributional grouping. In Portuguese, for instance, we can find, among many others:

`o` as Clitic or Definite Article

`a` as Definite article, Preposition or Clitic

`se` as Clitic or Conjunction

`embora` as Conjunction or Adverb

`só` as Adverb or Adjective

`português` as Adjective, Proper Noun or Common Noun

`bateria` as Common Noun or Verb

`como` as Verb, Adverb, Relative Pronoun, Interrogative Pron., Conjunction or Preposition

This implies that many lexeme-types are associated with more than one POS tag in the lexicon and that the single correct tag for each of their lexeme-tokens in a text has to be decided given the specific occurrence and context at stake.

From a computational point of view, the non trivial issue with respect to POS tagging consists thus in designing successful algorithms able to decide for each token of a lexeme in a text, and from the set of admissible POS tags for its type in the lexicon, which tag is the correct one to be assigned to that lexeme in that specific occurrence. Though apparently simple when presented under these terms, POS tagging is a very important phase in natural language processing. It handles a considerable amount of the ambiguity in utterances thus permitting to prune many worthless alternatives in the search space at a quite early stage of processing, even before the subsequent, and computationally expensive stages of syntactic and semantic processing.

The application of machine learning techniques to natural language POS tagging has matured to a point where it is now very rapid to develop new, state-of-the-art accuracy taggers (cf. Samuelsson and Voutilainen, 1997; Brill, 1995; Rathnaparkhi, 1996; Brants, 2002 a.o.). Provided that the training data is ready, obtaining a new tagger may be as rapid as a few seconds with some applications. Given the general-purpose of these techniques, this holds true for every language that they have been tried upon even though most of the initial research has been conducted over data from English. Accordingly, and letting aside the time required to accurately annotate the training corpus, the bulk of the time span needed to prepare a new tagger is determined basically by the time needed to prepare tools to handle language-specific issues. Such issues are found in each of the three major steps involved in the automatic tagging sensu latu of raw text, namely chunking, tokenizing, and tagging sensu stricto.

In the present paper, we report on solutions we arrived at for the specific issues that arise when developing a new automatic tagger for Portuguese and that are generic enough to be further reused to develop other new taggers for this language, possibly from other training data.

Chunker

When aiming at the tagging of raw text, the first processing phase consists in sentence chunking, by means of which the boundaries between sentences and paragraphs are marked and therefore sentential tokens are identified.

As in other languages with conventions similar to those adopted by the Portuguese orthography, in the vast majority of cases, a few designated punctuation symbols are used to mark the end of sentences. This set of terminator symbols includes the period '.' for declaratives, the question mark '?' for interrogatives, the exclamation mark '!' for exclamatives or imperatives, and the ellipsis '…' for sentences ending in some form of ellipsis or expectation. On the other hand a designated orthographic clue is also used to mark the beginning of a subsequent sentence, viz. a capital letter or a digit as the first character of the sentence.

Given these orthographic conventions, most sentence boundaries can be easily detected. The chunking algorithm just has to look for a sequence with one of the above terminators followed by a blank and a sentence starter. As in other languages

with similar conventions, there remain however some non-trivial cases to solve given the concomitant ambiguity of some terminator symbols and starting clues.[1]

The period is type-ambiguous between marking the end of a sentence or the end of an abbreviation. In the following example, the first token of the period symbol (in `Pav.`) marks the end of an abbreviation, and the second and third tokens mark the end of sentences.

```
A cantora Ágata actuou no Pav. Atlântico, a 15 de Março. 250
mil pessoas aplaudiram de pé.
```

On the other hand, a capital letter is type-ambiguous between marking the beginning of a sentence or of a proper noun. In the example above, the first token of a capital letter marks the beginning of the first sentence, and the second token signals the beginning of a proper noun. Also, a starting digit of a number can signal just the beginning of that number (as in `15` above) or the beginning of a sentence (as in `250` above).

Non-trivial cases occur when the token-ambiguity of both terminator and starter symbols is viable. For starter symbols, token-ambiguity is always available while for the period symbol such ambiguity can be found basically in two cases: (i) Typesetting conventions for Portuguese reject the occurrence of two consecutive periods: the form

```
A cantora Ágata actuou no Pav. Atlântico, a 15 de Mar.
```

is thus preferred to the form

```
A cantora Ágata actuou no Pav. Atlântico, a 15 de Mar..
```

This implies that any period used in an abbreviation may be also marking the end of a sentence. (ii) The string making part of the abbreviated word can itself be a word, as it happens for instance with sequences like `par.` (`par` followed by period or abbreviation of `parágrafo`) or `ter.` (`ter` followed by period or abbreviation of `terça-feira`).

Given that conventions very similar to these are used in languages other than Portuguese, and in particular in English, this sort of issues has been addressed in the literature and different solutions have been proposed for them[2]. Hence, they will not be in the focus of the present paper. Here we will rather address conventions for sentence bounding that are specific to Portuguese, or at least not found in other close Romance languages or English under the same format. Such conventions involve the marking of paragraph (turn taking) and sentence boundaries in written dialogue.

The two basic constraints for the format of a written dialog are that each character's turn appears in its own paragraph and that each utterance corresponds to a sentence, possibly with narrator's asides as parentheticals.

---

[1] Some harder cases involve the determination of sentence/paragraph boundaries indicated by starters of enumerated lists and quotation delimiters. This is not addressed in the present paper as Portuguese basically follows the same conventions as used in other languages like English, French etc.

[2] For a recent study and references cited therein, see Mikheev, 2002.

Focusing on the first sentence in a paragraph containing a character's turn, its beginning can be easily handled as it is marked with a dash ('–') immediately followed by the usual sentence starters:

```
<s> – Bom dia! </s>
```

Things get convoluted when it comes to narrator's asides. The beginning of a narrator's aside cannot initiate an utterance. It is always indicated by a dash and its ending is indicated by a dash if the aside does not conclude the sentence, or by a period if it is the last part of its sentence:

```
<p><s> – Apetece-me ir ao cinema – anunciou ele. </s></p>
```

```
<p><s> – Eu cá – disse ela – também quero. </s></p>
```

The fact that the preceding sentence has been concluded with a narrator's aside or not determines the way the beginning of the next utterance is marked. A character's sentence other than the first one in the current turn starts also with a dash if and only if it follows a sentence ending with a narrator's aside.

```
<p><s> – Vamos ao jardim. </s><s> Está um lindo dia. </s></p>
```

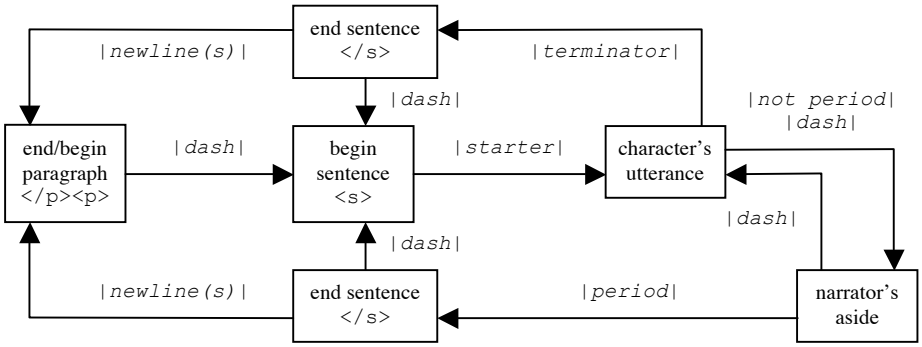
```
<p><s> – Não – replicou ela. </s><s> – Eu não vou. </s></p>
```

As for termination symbols of character's utterances, only those that are different from a period can appear before the beginning of a narrator's aside.

```
<s> – Bom dia! – exclamou ela. </s>
```

```
<s> – Mau dia – retorquiu ele, azedo. </s>
```

A perspicuous way of compiling and displaying the conventions related to written dialog orthographic format is by mean of a finite state automaton (FSA) – for a dialogue example illustrating these different conventions being applied, see Annex A. The states represent concepts such as "character's utterance" or "narrator's aside" and the transitions between these states are triggered by the occurrence of specific sequences of symbols in the input.



For example, the FSA above indicates that, when running over text, if the chunker is in a character's utterance, the occurrence of a sequence of two tokens separated by blank(s), where the first token is not a period and the second is a dash, will be taken as the start of the narrator's aside; on the other hand, if the chunker is in a narrator's

aside, the occurrence of a dash will be taken as the end of that aside and the continuation of the character's utterance that immediately preceded the aside.

Flex was used to implement this FSA. Flex can be viewed as being a superset of the C programming language specially suited for the creation of lexical analyzers. Through the use of regular expressions, the Flex syntax allows the programmer to easily specify which lexical patterns to look for in the input and, to each pattern, which action to trigger. For this procedure, we scored a recall of 99.94% and precision of 99.93% when tested on a 12 000 sentence corpus accurately hand tagged with respect to sentence and paragraph boundaries.

Tokenizer

For most tokens in a raw text, tokenization is a rather trivial task. After detaching punctuation marks that immediately follow lexemes without an intervening blank space, tokenization proceeds by taking advantage of the white space as a delimiter symbol indicating the boundary between two tokens:

`um exemplo ⇓ |um|exemplo|`

In view of subsequent accurate processing, especially in what concerns numbers, dates, amounts, etc., a careful tokenization should also mark spacing around punctuation or symbols. For instance, a solution we opted for was to separated every part of a number with a period or a comma as a single token:

`5.3 ⇓ |5|.|3|`

and to explicitly mark the occurrence of adjoining white spaces by inserting a designated symbol:

`1. 2 ⇓ |1|.□|2|`

`8 . 6 ⇓ |8|□.□|6|`

Turning now to issues that are specific in the tokenization of Portuguese text, there are a few trivial matters that can be also easily handled.

When following a form of verb `haver`, to which it is adjoined with an intervening hyphen, the preposition `de` should be detached as a single token and the hyphen removed:

`há-de ⇓ |há|de|`

Word endings marking alternative terminations should be explicitly acknowledged in view of future correct lemmatization:

`Caro(a) amigo(a) ⇓ |Caro|(a)|amigo|(a)|`

Clitic pronouns in enclisis should be detached from the verb:

`deu-se-lhes ⇓ |deu|se|lhes|`

If the enclisis of the pronoun induced a vocalic alternation, this should be explicitly marked by the tokenizer (cf. '#' in the example below) in view of future correct lemmatization:

vê-las ⇓ |vê#|las|

If the clitic pronoun appear in mesoclisis, it should also be detached, its original position signaled (cf. '-CL-' below) and the vocalic alternation marked as well:

afirmá-lo-ia ⇓ |afirmá#-CL-ia|lo|

Finally, contracted forms should also be handled. 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 special contracted forms with Personal Pronouns in accusative declination:

contigo (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 view of subsequent principled syntactic analysis, the tokenizer should thus expand these contractions as in the examples below:

do ⇓ |de|o|

pr'à ⇓ |para|a|

In the tokenization of Portuguese text, however, there are also non-trivial cases that present considerable difficulties for the design of the tokenization algorithm. Such cases involve type-ambiguous strings, i.e. strings that can be tokenized in more than one way:

deste ⇓ |deste| or deste ⇓ |de|este|

For an exhaustive list of type-ambiguous strings of Portuguese, and their frequency in our corpus, see Annex B.

In a general setup like ours, where one counts on a tagger trained over previously annotated data, this type of difficulties inevitably introduces some circularity: Although tagging decisions require that a previous tokenization process has been

completed, the tokenization of these ambiguous strings requires previous knowledge of the POS tag of the token(s) corresponding to the string. For instance, in the example above, `deste` would be tokenized as one token if and only if it had been tagged as a Verb, but for it to be tagged as a Verb it should have already been tokenized as one token.

In order to dissolve this circularity and correctly handle type-ambiguous strings, we used a two-level approach to tokenization where tagging is interpolated into the tokenization process, which proceeds now in two stages, one before and another after the tagger has been applied.

Accordingly, (i) a pre-tagging tokenizer definitely identifies every token except those related to ambiguous strings: These strings are provisionally identified as one token.

(ii) Subsequently, the tagger assigns a composite or a simple tag to every ambiguous string depending on it being 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. For instance, the string `deste` is tagged either a `deste_V` or as `deste_PREPDEM`.

(iii) Finally, a post-tagging tokenizer handles only ambiguous strings, breaking those that are tagged with a composite tag into two tokens and the corresponding tags.[3]

In order to implement the two-level tokenization approach just described, we used Ratnarparkhi's MXPOST system (Rathnarparkhi, 1996) to train a tagger for Portuguese. This system offers a state-of-the-art level of performance, having permitted to develop a tagger with 96.75% of success rate. The above approach was tested with the help of a 230 Ktoken hand annotated corpus, prepared from a corpus kindly granted by CLUL-Centro de Linguística da Universidade de Lisboa (Nascimento *et al*., 2000).[4] In this corpus, the ambiguous strings amount to 2% of the tokens.

This two-level tokenization approach permitted to successfully resolve 99.4% of these ambiguous cases, against a baseline of 78.2% of success. This baseline is obtained with the rough and ready heuristic of tokenizing every ambiguous string into two tokens, a heuristic straightforwardly suggested by the fact that 78.2% of the ambiguous strings are contractions in the test corpus.

Tagger

With suitable solutions for the Portuguese-specific issues concerning chunking and tokenization in place, the last step in the task of tagging raw text is the tagging procedure sensu stricto. That is, given that sentential and lexical tokens have been identified, the step yet to accomplished is to assign the POS tag to each lexical token, possibly taking into account the neighboring boundaries of the containing sentence or paragraph.

---

[3] For a detailed rendering of this solution, see Branco and Silva, 2003.

[4] We are very grateful to Fernanda Nascimento e Amália Mendes for their help.

For the development of the Portuguese tagger sensu stricto, we used the TnT software, kindly granted to us by Thorsten Brants (Brants, 2000). TnT is a statically based application to train taggers that, according to a recent overview (Giménez and Màrquez, 2003), offers the best features in terms of accuracy, for tagging, and in terms of efficiency, for tagging and training. It is based on second order Hidden Markov models supplemented with backoff via linear interpolation for smoothing purposes, and with suffix analysis for handling unknown words.

When using a machine-learning tool like TnT out of the shelf to develop a new tagger, the remaining critical issues dwell around the gathering of appropriate training data. For this purpose, we benefited from a 230 Ktoken corpus developed at CLUL referred to above. This corpus, containing a mix of excerpts from news and novels, was hand annotated with POS tags at CLUL in the scope of a theoretical linguistics research project aimed at studying clitic pronouns.

For the sake of the focusing on the language-critical issues involved in developing a new tagger, let us assume that one can rely on a previously annotated corpus as a starting point. Let us further assume that the consistency and accuracy of the annotation of such a general-purpose training corpus is ensured. The remaining concern is then directed towards manipulating and relabeling the training data in accordance with the tag set that needs to be opted for. The design of the tag set turns out thus to be the non-trivial, language-specific aspect that calls to be addressed.

In this respect, given that a statistically based application is being used, one finds the usual tension between increasing the discriminative power of the tagger — by using more tags — and minimizing the data sparseness — by using fewer tags. The search for the best performance of a POS tagger supported by a suitably tuned balance of these two attractors cannot be reduced, however, to arbitrarily playing around with the number and the assignment of tags. Syntactic categorization encodes basic linguistic generalizations about the distribution of lexemes, which by their own nature, are to be empirically uncovered, not superimposed in view of stipulative convenience.

By definition, a syntactic category identifies, under the same tag, tokens with identical syntactic distribution, i.e. tokens that in any occurrence receiving that tag, can replace each other while preserving the grammaticality of the corresponding linguistic construction, modulo the adoption of suitable subcategorization constraints impinging over them. If the goal is the development of a top-accuracy and linguistically principled tagger that optimally supports subsequent syntactic parsing, this is a criterion that one cannot lose sight of in the choice of the tag set.

Taking the preceding considerations into account, there are possible "candidate" categories or subcategories that should not to be included in the tag set used to annotate the corpus over which the tagger is to be trained.

In the first place, different tags not justified by different distribution are to be excluded. This is the case, for instance, of tags indicating the degree of an adjective (example: `alto_ADJNORM`, `altíssimo_ADJSUP`).

Tough conveying some distribution-related information, there may be tags that can be unequivocally inferred from the form of the token at stake. In view of decreasing the data sparseness, such tags should be avoided. For example, this is the case of tags indicating the polarity of an adverb (example: `sim_ADVPOS`; `nem_ADVNEG`), or tags

indicating inflectional features, which can be subsequently determined from suffixes by a lemmatizer (example: `alto_ADJMascSing`, `altas_ADJFemPlu`).

Also, when considering tag sets proposed in grammar textbooks of a more traditional, philological-oriented persuasion, it is not unusual to find categories aimed at indicating the constituency status of the phrase containing the relevant token. Such different tags encode information about whether the token at stake is a constituent of an elided or of a non-elided phrase but not an actual difference with respect to the syntactic distribution of that token. One example of this is the category "indefinite pronoun" versus some other category of closed classes. This category has been proposed for tagging articles, demonstratives or other pronominal items in headless Noun Phrases. For instance, according to such traditional views, the demonstrative `aquele` would receive DEM in the non-elliptical NP in `li [aquele_DEM livro]`$_{NP}$ but it would receive INDPRON in the corresponding elliptical NP in `li [aquele_INDPRON Ø]`$_{NP}$). Given that no difference with respect to syntactic distribution of items like `aquele` is at stake, and in view of taming the data sparseness effect, the tags indicating the elliptical status of the containing phrase have no place in our tag set. Returning to the specific examples above, `aquele` receives the same tag on both cases and the last example is tagged as: `li [aquele_DEM Ø]`$_{NP}$

Under more traditional approaches, single-word NPs like `tudo` are also proposed to receive the "indefinite pronoun" tag or a similar one. It goes without saying that a tag like `IN` (Indefinite Nominals), for instance, should be included in the tag set to cover these cases.

It is of note that the rationale discussed above and followed to circumscribe the tag set, not only helps to exclude possible candidate tags, but also to isolate and include categories that are usually not taken into account in a more traditional perspective.

Though being verbal forms, gerund, past participle and infinitive forms each have a distribution of its own due to the fact that they are the main predicators of subordinate clauses with specific distribution. Moreover, infinitival forms support nominative constituents (e.g. `[ouvir_INF música]`$_{NP}$ `diminui o stress`) and past participle can be used with adjectival force (e.g. `o candidato eleito_PTP não chegou a tomar posse`). The tags `GER`, `PTP` and `INF` are thus included in the tag set to enhance the discriminative power of the tag.

Other "non-canonical" tags are also included. These may be less interesting from a general linguistic point of view but they are important to improve also the contribution of the tagger for subsequent processing stages, e.g. named entity recognition. They cover dialogue particles (`adeus`, olá,…) social titles (Pres., Drª., prof.,...), part of addresses (Rua, Av., Rot.,...), email addresses, months (Janeiro, jan.,…), days of the week (Terça-feira, ter., Quarta,…), measurement units (km, kg, b.p.m.,...) as distinct syntactic classes. Our tag set includes also specific tags for digits, roman numerals, denominators of fractions (meio, terço, décimo, %,...), orders of magnitude (centenas, biliões,…), symbols (/, #,…) and letters.

Finally, in order to tag multi-word expressions from closed classes, a special tagging scheme is used where each component word receives the same tag prefixed by `L`, and followed by the corresponding index number. The following are some examples of the application of this scheme:

```
apesar_LPREP1 de_LPREP2
```

```
a_LCJ1 fim_LCJ2 de_LCJ3

a_LADV1 tempo_LADV2 e_LADV3 horas_LADV4
```

With the full tag set for the training data (cf. Annex C) defined under the above guidelines, we prepared the training corpus by converting and adjusting the initial tagged corpus from CLUL. With these data and the help of the TnT tool, a tagger for Portuguese was trained. When coupled with a chunker and a tokenizer implemented along the lines discussed in the Sections above, it scored 97.2% accuracy. This figure was obtained with one run test over a held out evaluation corpus with the 10% not used for training. This result is in line with the state-of-the-art performance reported for German (96.7%) or English (96.7%) with the same tool over, respectively, the NEGRA Corpus (320 Ktokens) and the Penn Treebank (1.2 Mtokens) corpora, and an accuracy measurement averaged over 10 test runs (Brants, 2000).

Concluding remarks

The application of machine learning techniques to natural language POS tagging permits to develop new taggers with state-of-the-art accuracy very rapidly. Although most of the initial research in this topic has been conducted over data from English, given the broad coverage of these techniques, this holds true for every language they have been tried upon. Provided that the training data is ready, obtaining a new tagger may be as rapid as a few seconds with the most efficient applications, like the TnT software used in the research results reported above. Disregarding the time required to hand annotate the training data, the time span needed to prepare a new tagger turns out to be determined basically by the time needed to prepare the tools aimed at handling language-specific issues.

These language-specific issues are to be found in each of the three major steps involved in the automatic tagging sensu latu of raw text: chunking, tokenizing, and tagging sensu stricto. As for Portuguese, the specific issues are: in terms of chunking, the orthographic conventions for written dialog; in terms of tokenizing, strings type-ambiguous between one or two tokens; finally, in terms of tagging sensu stricto, the compilation of the tag set for training the tagger.

In this paper, we presented solutions for these specific issues. These solutions are generic enough to be reused and thus to further reduce the time span required to develop taggers for Portuguese. We also presented evaluation results showing that, when coupled together to form a tagger for raw text, these solutions do not degrade overall accuracy and efficiency, keeping up with state-of-the-art performance.

References

Branco, António Horta and João Silva, 2003, "Contractions: breaking the tokenization-tagging circularity", In Mamede et al. (eds.) *Computational Processing of the Portuguese Language*, Berlin: Springer, LNAI 2721, pp.167—170.

Brants, Thorsten, 2000, “TnT - A Statistical Part-of-speech Tagger”. *Proceedings of the Applied Natural Language Processing (ANLP)*, Association for Computational Linguistics, pp. 224—231.

Brill, Eric, 1995, “Transformation-based Error-driven Learning and Natural Language Processing: A case study in part-of-speech tagging”. *Computational Linguistics*, 21, pp. 543—565

Giménez, Jesús and Lluís Màrquez, 2003, “Fast and Accurate Part-of-Speech Tagging: The SVM Approach Revisited”. *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP)*, Sofia: Bulgarian Academy of Sciences, pp. 158—165.

Mikheev, Andrei, 2002, “Periods, Capitalized Words, etc.” *Computational Linguistics* 28(3), pp. 289—318.

Nascimento, Fernanda, Luísa Pereira and João Saramago, 2000, "Portuguese Corpora at CLUL" *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC)*, Paris: ELRA, pp. 1603—1607.

Rathnaparkhi, Adwaith, 1996, “A Maximum Entropy Part-of-speech Tagger”. *Proceedings of the Empirical Methods on Natural Language Processing (EMNLP)*, Association for Computational Linguistics, pp. 133—142.

Samuelsson, Chris and Atro Voutilainen, 1997, “Comparing a Linguistic and a Stochastic Tagger”. *Proceedings of Annual Meeting of the Association for Computational Linguistics (ACL)*, Association for Computational Linguistics, pp. 246—253.

Annex A – Showcase example for dialogue orthography

<s> — Ficamos no jardim. </s><s> Está um lindo dia. </s>
<s> — Quero ir ao cinema! — anunciou o gémeo. </s>
<s> — Eu cá — disse a gémea — também quero — e olhou para a mãe. </s>
<s> — … não — hesitou a mãe. </s><s> — Ficamos aqui. </s>

Annex B – Type-ambiguous strings

Ambiguous Strings	**Occurrences in our test corpus (230 Ktokens)**		
	Total	**As one token**	**As 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%

Annex C

Tag	Category	Examples
ADJ	Adjectives	bom, brilhante, eficaz
ADV	Adverbs	hoje, já, sim, felizmente
CARD	Cardinals	zero, dez, cem, mil
CJ	Conjunctions	e, ou, nem
CL	Clitics	o, lhe, se
CN	Common Nouns	computador, cidade, ideia
DA	Definite Articles	o, os
DEM	Demonstrative Pronouns	este, esses, aquele
DFR	Denominators of Fractions	meio, terço, décimo, %
DGTR	Roman Numerals	VI, LX, MMIII, MCMXCIX
DGT	Digits	0, 1, 42, 12345, 67890
DIAG	Dialogue Particles	adeus, olá, alô
EADR	Electronic Addresses	http://www.di.fc.ul.pt
GER	Gerunds	afirmando, sendo, vivendo
IA	Indefinite Articles	um, uns
IN	Indefinite Nominals	tudo, alguém, ninguém
INF	Infinitives	amar, correr, ser
INT	Interrogative Pronouns	quem, como, quando
ITJ	Interjections	oh, ah, eh
LTR	Letters	a, b, c
MGT	Magnitude Classes	unidade, dezena, dúzia, resma
MTH	Months	Janeiro, Dezembro
NP	Noun Phrases	idem
ORD	Ordinals	primeiro, centésimo, penúltimo
PADR	Part of Address	Rua, av., rot.
PNM	Part of Name	Lisboa, António, João
PNT	Punctuation Marks	., ?, (
POSS	Possessive Pronouns	meu, teu, seu
PP	Prepositional Phrases	algures
PREP	Prepositions	de, para
PRS	Personal Pronouns	eu, tu, ele
PTP	Past Participles	sido, afirmado, vivido
QD	Quantificational Determiners	todos, muitos, nenhum
REL	Relative Pronouns	que, quem, cujo
STT	Social Titles	Presidente, dr^{a}., prof.
SYB	Symbols	@, #, &
TERMN	Optional Terminations	(s), (as)
UNIT	Measurement Units	km, kg, b.p.m.
V	Verbs (other than PTP or GER)	ser, afirmar, viver
WD	Week Days	segunda, terça-feira, sábado
Tags for multi-word expressions		
LTAG1...LTAGn	Each token *i* of a *n*-word expression of category *TAG* gets L*TAGi*	
LCJ1...LCJ4	4-token conjunctional expression	de modo a que
LPREP1 LPREP2	2-token prepositional expression	apesar de
LADV1...LADV3	3-token adverbial expression	no entanto
...		