Shallow Processing of Portuguese
From Sentence Chunking to Nominal Lemmatization

João Ricardo Silva and António Branco

University of Lisbon
{jsilva,antonio.branco}@di.fc.ul.pt

Abstract The work described in my MSc dissertation [1], carried out under the supervision of Prof. António Branco, proposes a set of shallow processing procedures for the computational processing of Portuguese. Five tasks are covered, namely Sentence Segmentation, Tokenization, Part-of-Speech Tagging, Nominal Featurization and Nominal Lemmatization. For each task, I begin by identifying and describing the key problems that it raises, with special focus on those problems that are specific to Portuguese. After an overview of existing approaches and tools, I describe the solutions I follow to tackle the issues raised previously. Finally, I report on my implementations of these solutions, which are found either to produce state-of-the-art performance or, in some cases, to advance the state-of-the-art. The major result of my dissertation is thus threefold: A description of the problems found in NLP of Portuguese; an overview of current approaches and tools; and a set of algorithms to tackle those problems, together with their evaluation results.

1 Introduction

A major difficulty in processing natural language lies in the handling of ambiguity. This is a problem that arises in every stage of processing, and ranges from semantic ambiguity that requires real-world knowledge to resolve, to low-level—though not necessarily easier—issues such as deciding whether a particular occurrence of the period symbol marks the end of a sentence or is part of an abbreviation.

If left unresolved, ambiguity will accumulate: An ambiguous linguistic construct will cause a branching into different possible cases which, in latter stages of processing, can also be specified into their own ambiguous cases. Therefore, practical systems have a great need of curbing this explosion of alternatives.

Shallow processing (SP) is an approach to NLP that gained popularity as an efficient way of mitigating the problem caused by ambiguity, and delivering results that are useful without resorting to more complex and expensive syntactic and semantic processing.

The core rationale of SP is straightforward: It associates linguistic information to text based on local information (i.e. using just the word itself or perhaps a very limited window of context).

As a consequence of using only local information, some ambiguous constructs can be resolved quickly (often, as soon as one presents itself). Even if ambiguity cannot be fully resolved, the problem space can sometimes be reduced by restricting the amount of possible alternatives. In either case, the combinatorial growth of alternatives that is caused by ambiguity is greatly reduced.

To achieve this, SP tasks tend to be highly specialized because they need to address specific different cases of ambiguity. So, instead of an all encompassing tool, SP uses tokenizers, stemmers, named entity recognizers, multi-word expression detectors, noun phrase chunkers, word-sense disambiguators, prepositional phrase attachment resolvers, etc. For the same reason, SP often combines statistical and symbolical approaches, allowing for a much better adaptation of each tool to the specific task that is to be handled.

Main Results. The main results of my dissertation are: (i) The identification and description of the key problems raised by each of the tasks that are covered, with special focus on those problems that are specific to Portuguese; (ii) an overview of current approaches and tools to tackle those tasks; and (iii) the description, implementation and evaluation of state-of-the-art, shallow processing algorithms to solve those problems.

2 The Tasks

My dissertation covers five NLP tasks: (i) Sentence Segmentation, (ii) Tokenization, (iii) Part-of-Speech Tagging, (iv) Nominal Featurization and (v) Nominal Lemmatization.

This particular set of tasks was chosen because there is an inherent "pipeline" sequence to the tasks that are performed in NLP. For instance, morphological analysis of nominal tokens requires a previous step of morphosyntactic tagging that identifies nominal expressions, which in turn requires an already tokenized text. The five tasks above can be seen as the initial steps in that pipeline.

The corpus. For the tools to be evaluated—and for them to be trained, when applicable—there has to be a properly prepared corpus. The corpus used throughout this dissertation was prepared from a ca. $260,000$ token corpus, composed mostly by excerpts from newspapers and fiction novels.

It is important to note that this corpus evolved alongside the tools. For instance, its original POS annotation was changed to conform to the tagset used in this dissertation and new annotation layers (inflection features, lemma, etc.) were added, and manually checked by trained linguists, as the corresponding tools were developed. As a consequence, at the time many of the tools in this dissertation were being developed, the corpus was not annotated in a way that allowed training machine-learning approaches.

2.1 Sentence Segmentation

The sentence is the maximum syntactic element in natural languages. The practical implications of this is that one must know where each sentence begins and ends to be able to analyze it syntactically. However, in computer terms unprocessed text is simply a string of characters, with no marked boundaries for sentences. It is the task of a sentence segmenter to detect and mark the boundaries of each sentence and paragraph.

For most cases, sentence segmentation is easy since in Portuguese, as well as in other languages with similar orthographic conventions, the ending of a sentence is marked by the use of a punctuation symbol from a known set, such as . (period), ? (question mark), ! (exclamation mark) or ... (ellipsis).

There are, however, harder cases which must be resolved, including the ambiguous use of the dash symbol (particularly in dialog) and the handling of the ambiguity pertaining to the period symbol.[1]

With respect to the dash symbol, the difficulty stems from the ambiguity caused by using the dash for various purposes. In dialog, this symbol is used to mark (i) the beginning of a speaker's turn, (ii) the beginning of a narrator's aside or (iii) the retaking of a speaker's turn after a narrator's aside. In addition, the dash may also be used outside of dialog for delimiting a parenthetical expression.

As for the period symbol, it is also ambiguous when abbreviations are involved since, when an abbreviation is followed by a word starting with a capital letter, there is an ambiguity between the period marking the end of the sentence or marking an abbreviation within a sentence.

The Sentence Segmentation task was implemented with a finite-state automaton since, at the time work was started, the corpus I was using was not annotated with sentence and paragraph boundaries, which precluded a machine-learning approach.

The tool was then used to provide an automatically segmented corpus that, after manual correction, was used for evaluation. For this task, I define recall as the proportion of boundaries that were correctly found and precision as the proportion of emitted boundaries that the segmenter got right.

The sentence segmentation tool scored 99.94% recall and 99.93% precision over the ca. $12,000$ sentences in the corpus. This can also be seen in terms of error rates, with the segmenter having missed 0.06% boundaries and having produced 0.07% extra boundaries.

2.2 Tokenization

A tokenizer takes a string of characters and delimits the tokens that it includes, such as words, punctuation marks and other symbols. In this respect, tokenization can be seen as a normalization step through which we enforce that each

[1] There are also hard cases involving quotations, sentence wrapping and headings that will not be addressed in this paper.

token be delimited in the same way. This consists of delimiting words, separating punctuation from words, expanding contractions into their components, detaching clitics from verbs, etc.

There are several non-trivial issues regarding tokenization, but in this article I will focus my attention only in the problem caused by token-ambiguous strings.

These are strings that can be tokenized as one or as two tokens, depending on their occurrence. For instance, `deste` can be tokenized as a single token if it is an occurrence of a form of the verb `dar`,[2] or as two tokens if it is an occurrence of the contraction of the Preposition `de` and the Demonstrative `este`.

Even though there are only 14 such strings in Portuguese,[3] handling these cases correctly is of critical importance as they amount to 2% of the tokens in the corpus. Moreover, since these strings and their expansions correspond mostly to words from the functional categories, it is expected that errors in their tokenization would cause a considerable degradation of accuracy for subsequent processing stages.

The difficulty in handling these cases arises from the fact that the decision on how to tokenize an ambiguous string depends on its morphosyntactic category. However, at this stage of processing, morphosyntactic categories have not yet been assigned, as that task requires a previous tokenization step. One is thus confronted with a problem of circularity: To properly handle token-ambiguous strings, one needs morphosyntactic tags; but before assigning morphosyntactic tags one expects an already tokenized text.

To break this circularity, I follow a two-stage approach to tokenization which also can be envisaged as POS tagging being interpolated into the tokenization process.

The basic rationale is as follows: Tokenization proceeds as normally, except that every token-ambiguous string is temporarily tokenized as a single token. After that step, the POS tagger runs over the result, assigning a POS tag to every token. Finally, a post-tagging tokenizer searches specifically for the ambiguous strings and tokenizes them according to the POS tag they received.

To achieve this, the corpus that is used for training the POS tagger must be adapted. In this corpus, the token-ambiguous strings are always tokenized as a single token but receive a portmanteau tag when they correspond to a contraction. For instance, all occurrences of the token-ambiguous string `deste` are tokenized as a single token, but they receive the portmanteau tag `PREP+DEM` when they are an occurrence of the contraction (and, per usual, receive the tag for Verb, when occurring as such).

After the tagger is run, a second tokenization stage, the post-tagging tokenizer, looks for token-ambiguous strings tagged with portmanteau tags and expands them.

Using the state-of-the-art POS tagger that is described in the next Section, the tokenizer implemented achieves 99.44% accuracy in the tokenization of ambiguous strings.

[2] *Pretérito Perfeito* tense, second person, singular.

[3] See [1, p. 20] for a full list.

2.3 Part-of-Speech Tagging

One of the most well studied tasks in NLP is that of part-of-speech tagging, where tokens are automatically classified into morphosyntactic categories, or POS, such as Common Noun, Adjective, Verb, Preposition, Conjunction, etc. according to their context of occurrence.

The main non-trivial issues that must be resolved for this task are (i) the tagging of those words that are lexically ambiguous with respect to POS, (ii) the selection of the POS tagset will be used and (iii) the handling of categories that span more than one token (multi-word expressions). Again, in this article I will focus only one of these issues, namely that of multi-word expressions.

In order to tag multi-word expressions, a special tagging scheme is used where each component word in an expression receives the same tag (corresponding to the POS of the whole expression) prefixed by `L`, and followed by an index number corresponding to its position within the expression. For instance, the multi-word `apesar de` (*Eng. despite*), a Preposition (`PREP`), receives `apesar/LPREP1 de/LPREP2`.

Given the widespread availability of a large number of machine-learning, language-independent POS taggers, I used third-party tools—namely, Brill's TBL [2], Brant's TnT [3] and Ratnaparkhi's MXPOST [4]—instead of implementing a POS tagger from scratch.

The performance of the taggers was assessed using 10-fold cross-evaluation: The taggers were trained over 90% of the corpus and evaluated over the remaining 10%. This was repeated over 10 runs—each with a different partition of the corpus—and the results were averaged. The taggers were trained and run using their default settings. The evaluation results are summarized in Table 1.

TBL	TnT	MXPOST
97.09%	96.87%	97.08%

Table 1. POS tagger accuracy

The scores that were obtained are in line with the current state-of-the-art results for POS tagging and, in particular, with the results reported for those same three tools for other languages. In addition, when comparing with other POS taggers for Portuguese, one finds that these seem to currently be the best shallow processing POS taggers for Portuguese [1, p. 94].

2.4 Nominal Featurization

Nominal featurization is the process of associating inflectional and derivational feature values to tokens of the nominal categories: Adjective, Common Noun and others that bear the same features (Article, Determiner, etc.) The key difficulties found in this task depend on the approach that is opted for.

For a stochastic approach, the data-sparseness will be the key issue since the inclusion of morphological information will require a much larger tagset, leading to a lower tagging precision due to the well known sparseness of data bottleneck: For the same amount of training data, a larger tagset will lead to the existence of more parameters for which there is no significant data available.

For a lexicon-based approach the ambiguity caused by lexically unknown and by invariant words will come to be the critical problem. The former are not present in the lexicon while the latter, the so-called "invariant" words because they are lexically ambiguous with respect to inflection features, cannot be assigned the relevant feature values through a lexical look-up alone.

The morphological regularities found in Portuguese suggest a straightforward rule-based algorithm for the autonomous assignment of inflection feature values—even to unknown words—given that word terminations are typically associated with a default feature value.[4]

To handle invariant words, an algorithm can be envisaged that builds upon the fact that there is Gender and Number agreement in Portuguese, in particular within Noun Phrases (NPs): The Gender and Number feature values for the Common Noun in a NP determine the agreement of Determiners, Quantifiers, etc. within the same NP.

Consequently, we can determine the inflection feature values for an invariant Common Noun if we know the inflection feature values of other tokens with which the Common Noun agrees.

The procedure can be outlined as follows: All words from the closed classes that have inflection features (Demonstrative, Determiner, Quantifier, etc.) are collected together with their corresponding inflection tags.[5] During inflection analysis of a text, the inflection tags assigned to words from these closed classes are propagated to the words from open classes (Adjective and Common Noun) that immediately follow them. These may, in turn, propagate the received tags to other words.

The main difficulty in applying this mechanism is in ensuring that that feature propagation occurs only within NP boundaries and that it does not cross into other phrases found embedded in the NP. For this effect, some patterns of tokens and POS tags are defined such that, when they are found, tag propagation is prevented from taking place. For more detail on this issue, refer to [1, p. 102].

The Nominal Featurization tool achieves a f-score of 97.03% when running over correct POS.

2.5 Nominal Lemmatization

Lemmatization is the process which associates canonical, inflectionally normalized forms (the lemmas) to tokens. The major difficulty that is found in this task

[4] Any exceptions to this can be easily found by searches in machine-readable dictionaries. The list of exceptions tends to be rather stable since new words that enter the lexicon usually follow the rule instead of being exceptions.

[5] This can be done since the words from the closed classes form a closed list.

is that the lemma of a given word type may depend on the sense of its relevant token.

In some cases, such as with the word `ética`, the ambiguity can be resolved by knowing the POS of the token. For example, `ética`, when occurring as an Adjective, is the feminine singular form of *ethical* and should therefore be lemmatized into `ético`, the masculine singular form. However, when occurring as a Common Noun, `ética` has the meaning of *ethics*, and its lemma is `ética` instead.

There are cases, however, where different occurrences of a type, though bearing the same POS tag, can nonetheless receive different lemmas. For instance, the word `copas` may refer to the *hearts* suit of playing cards, in which case its lemma is `copas`, or it may be the plural form of `copa` (*Eng.: cupboard, treetop*), in which case it should be lemmatized into the singular form `copa`.

Note that ambiguous words such as `copas` cannot be resolved by any lemmatization process that is applied before a word sense disambiguation (WSD) stage. In fact, the existence of these words presents an inevitable upper bound for the shallow nominal lemmatization process, preventing this process from ever achieving total coverage of the targeted set of word forms.

To implement the nominal lemmatizer, I build upon the morphological regularities found in word inflection and use a set of transformation rules that revert to the form of the lemma.

The basic rationale is thus to gather a set of transformation rules [1, p. 159] that, depending on the termination of a word, replace that termination by another, and complement this set of rules with a list of exceptions collected through the use of machine-readable dictionaries that allow searching for words on the basis of their termination. Neologisms are expected to comply with the regular morphology and are thus accounted for by the rules. This basic algorithm is extended to allow for the recursive application of transformation rules, for handling non-inflectional affixes and for a branching search for lemmas that allows for several transformation to be tested in parallel.

The Nominal Lemmatization tool achieves a f-score of 98.73%.

3 Final Remarks

The tasks that are discussed in my dissertation are some of the initial stages in NLP. In any of these tasks we find major difficulties that are invariably caused by ambiguity. The identification of these key issues is one of the contributions of my dissertation.

I found shallow processing to be an efficient way of dealing with the problems caused by ambiguity: As a consequence of only using local information and specialized approaches—be them rule-based, stochastic or a combination of both,—it can resolve ambiguity without needing complex syntactic and semantic processing, and still deliver useful results.

Each of the tasks was addressed through a shallow processing algorithm that achieved—and in some cases advanced—the state-of-the-art.

Several systems are now supported by the pipeline of shallow processing tools described in my dissertation. In particular, a POS tagger, a named-entity recognizer and a question-answering system. Services demonstrating these systems can be reached through `http://lxcenter.di.fc.ul.pt/`.

The dissertation, published as Technical Report DI-FCUL-TR-07-16, may be found at the following address: `http://hdl.handle.net/10455/3095`.

References

1. Silva, J.: Shallow processing of Portuguese: From sentence chunking to nominal lemmatization. Master's thesis, University of Lisbon (2007) Published as Technical Report DI-FCUL-TR-07-16 at `http://hdl.handle.net/10455/3095`.
2. Brill, E.: Transformation-based error-driven learning and natural language processing: A case study in part-of speech tagging. Computational Linguistics **21**(4) (1995) 543–565
3. Brants, T.: TnT — a statistical part-of-speech tagger. In: Proceedings of the 6th Applied Natural Language Processing Conference and the 1st North American Chapter of the ACL. (2000) 224–231
4. Ratnaparkhi, A.: A maximum entropy model for part-of-speech tagging. In Brill, E., Church, K., eds.: Proceedings of the 1st Conference on Empirical Methods in Natural Language Processing, ACL (1996) 133–142