LemPORT: a High-Accuracy Cross-Platform Lemmatizer for Portuguese

Ricardo Rodrigues, Hugo Gonçalo Oliveira, and Paulo Gomes

Centre for Informatics and Systems of the University of Coimbra
Pinhal de Marrocos, Coimbra, Portugal
{rmanuel,hroliv,pgomes}@dei.uc.pt

Abstract

Although lemmatization is a very common subtask in many natural language processing tasks, there is a lack of available true cross-platform lemmatization tools specifically targeted for Portuguese, namely for integration in projects developed in Java. To address this issue, we have developed a lemmatizer, initially just for our own use, but which we have decided to make publicly available. The lemmatizer, presented in this document, yields an overall accuracy over 98% when compared against a manually revised corpus.

1998 ACM Subject Classification I.2.7 Natural Language Processing

Keywords and phrases lemmatization, normalization, rules, lexicon

Digital Object Identifier 10.4230/OASIcs.SLATE.2014.267

1 Introduction

Almost every task related to natural language processing (NLP) [10] must apply some kind of text normalization. Texts must be split into sentences, and sentences into tokens (words and punctuation). Words must also be further processed in order to facilitate their analysis: for instance, when searching for specific words, as it happens in many information retrieval systems (IR) [2], their inflections must also be considered in order to broaden the results.

The most common approaches for tackling this problem and collapsing morphological variants of the same word are: (i) stemming, which essentially consists of stripping off word endings; and (ii) lemmatization, where words with the same morphological root are identified, despite their surface differences.

Stemming is easier and faster to implement, but discards potentially useful information, by making it virtually impossible to distinguish a verb from a noun or an adjective in its stemmed form. Moreover, the stem is not necessarily a recognizable dictionary word.

Lemmatization, in contrast, considers the syntactic category of words, presenting, for instance, different lemmas for a noun or a verb (in the same word family). This nuance is practically lost in English, where the same lemma can assume multiple syntactic categories, but it is of paramount importance in romance languages, including Portuguese.

In the remaining sections of this document, we make a brief contextualization on lemmatization and related work, then proceed to describe our method, followed by the evaluation performed and results obtained, after what we end up drawing some conclusions and pointing future paths for eventual improvement.

2 A Brief Contextualization

In some situations, lemmatization and stemming operate in a similar way: given a set of affixes, for each word in a list (a phrase, a sentence or a text), check if the word ends with

3^{rd} Symposium on Languages, Applications and Technologies (SLATE'14).
Editors: Maria João Varanda Pereira, José Paulo Leal, and Alberto Simões; pp. 267–274
OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

any of the affixes, and, if so, and apart from a few exceptions, remove the affix from the word. The problem is that this process is sometimes not enough to retrieve the dictionary form of a word, it is too disruptive and, in most cases, the stem is not the same as the lemma. Next, we point out how lemmatization should perform, and discuss related work.

2.1 The Lemmatization Process

In order to retrieve the lemma of a word, it is sometimes enough to remove the word's affix. This typically happens in *noun number normalization*, which is the case of `carros` losing the trailing `s` and becoming `carro` (`cars` → `car`). In other cases, such as in *noun gender normalization*, it is necessary to replace the affix. For instance, the dictionary form of `gata` (a female cat) is `gato` (a male cat), which requires replacing the feminine affix, `a`, for the masculine affix, `o`. The same applies to *verb normalization*: for instance, the lemmatized version of `[eu] estudei` (`[I] studied`) becomes `[eu] estudar` (`[to] study`), replacing the verbal inflection affix with the associated conjugation infinitive affix (`ar`, `er`, or `ir`).

For Portuguese, lemmatization may include the following types of normalization: noun (gender, number, augmentative and diminutive), adjective (gender, number, augmentative, diminutive, and superlative), article (gender and number), pronoun (gender and number), proposition (gender and number), adverb (manner) and verb (regular and irregular). Proper nouns, numbers, interjections and conjunctions are usually ignored.

Determining what kind of normalization to apply depends on the syntactic category, or part-of-speech (POS) tag, of each word. The task of identifying the POS tag of a word is out of our scope, and we resort to a freely available solution: the OpenNLP Library[1], by the Apache Software Foundation, with some minor tweaks. So, a lemmatizer must take as input both word and POS tag to produce the coveted lemma.

In theory, knowing the syntactic category of a word and the rules to normalize it, lemmatization should be a straightforward process. In practice, although these rules cover the vast majority of the cases in any given text, exceptions to these same rules defeat the goal of reaching an accuracy close to 100%. Moreover, the exceptions usually happen to be found in the oldest and most used lexemes of any lexicon. For instance, the verbs *to be* and *to have* are highly irregular in every western language, including Portuguese and English.

The same happens in other syntactic categories, such as nouns. For instance, the singular form of `capitães` (`captains`) is `capitão` (and not `capitãe*`). There are also cases where the masculine form of a noun is quite different from the feminine version. And the most problematic case is perhaps when a word is already in its dictionary form but appears to be in an inflected form, when, in reality, it is not – as happens with `farinha` (`flour`), that seems to be a diminutive (due to its ending in `inha`, the most common diminutive affix).

Most lemmatization tools use a rule system (covering the vast majority of cases for each type of normalization), and specify exceptions to the rules, using, at some point, a lexicon for validating the lemmas produced, or for extracting rules (and their exceptions).

2.2 Related Work

Even though most researchers on Portuguese NLP must use some sort of lemmatization tool, those tools are hard to come by, at least as isolated lemmatizers, although there are suites of tools, including morphological analyzers, that produce lemmas as a part of the

[1] The OpenNLP toolkit can be found at `https://opennlp.apache.org`.

outcome. While useful, these suites and morphological analyzers leave to the users the task of post-processing the output in order to extract just the intended lemmas, imposing also some restrictions on previous processing, as it usually has to be done using the same tools.

Three such tools are known to exist, targeting specifically Portuguese: jSpell,[2] FreeLing,[3] and LX-Suite.[4] All have web interfaces, with jSpell and Freeling providing downloadable versions and source code. Moreover, jSpell is available as C and Perl libraries, as well as a MS Windows binary; and Freeling is available as a Debian package and also as a MS Windows binary, in addition to an API in Java and another in Python, along with the native C++ API.

Both jSpell and Freeling start with a collection of lemmas and use rules to create (all) inflections and derivations from those lemmas, alongside data such as number, syntactic category, or person (in the case of verbs) [12, 5]. It is the output of that process that is used to lemmatize words, matching them against the produced inflections and derivations, retrieving the originating lemma.

Regarding LX-Suite, though only a description of a nominal lemmatizer [3] has been found, which is believed to be the lemmatizer behind the LX-Suite [4] of NLP tools (belonging to the LX-Center), it uses a lexicon for the purpose of retrieving exceptions to the lemmatization rules: if the lexicon contains a word (therefore, a valid word) that would be processed by the rules (but should not), it is marked as an exception and added to the exception list. According to the authors, that lemmatizer achieved an accuracy of 97.87%, when tested against a hand annotated corpus with 260,000 tokens.

For jSpell and Freeling, although evaluations for the morphological analyzers do exist, no statement regarding specifically the accuracy of the respective lemmatizers was found.

3 Our Approach to Lemmatization

Our lemmatization method shares features with other approaches, such as the use of rules and, more recently, a lexicon. The way these resources are combined leads to a high accuracy value. In this section, we describe our method and its evolution.

3.1 The Use of Rules

In its earlier versions, our lemmatizer depended only on handmade rules and exceptions. Each of the normalization steps included in the lemmatization process had an associated set of rules. As such, there were rules for (and in this order):

1. manner (adverb) normalization;
2. number normalization;
3. superlative normalization;
4. augmentative normalization;
5. diminutive normalization;
6. gender normalization;
7. verb normalization (for regular and irregular verbs).

Each of these rules were associated to one or more POS tags. So, for instance, gender normalization could be applied to nouns and also to adjectives, while superlative normalization would only be applied to adjectives. The rules were defined by the target affix, the POS

[2] jSpell is freely available from `http://natura.di.uminho.pt/wiki/doku.php?id=ferramentas:jspell`.
[3] Freeling is freely available from `http://nlp.lsi.upc.edu/freeling/`.
[4] The LX-Suite can be found at `http://lxcenter.di.fc.ul.pt/services/en/LXServicesSuite.html`.

```
<replacement target="inha" tag="n|n-adj|adj" →
exceptions="azinha|...|farinha|...|sardinha|...|vizinha">a</replacement>
```

Figure 1 A rule for transforming a diminutive into its “normal” form.

```
<prefix>(a|ab|abs|...|sub|super|supra|...|vis).?\-?</prefix>
<suffix>[\wàáãâéêíóõôúç\-]*</sufix>
...
<replacement target="a[gj][aeiío]" tag="v|v-fin|v-ger|v-pcp|v-inf"> →
agir</replacement>
```

Figure 2 A rule for transforming a inflected verb into its infinitive form.

tags of the words they should be applied to, exceptions, and the replacement for the target affix. All rules were declared in XML files, illustrated by the example in Fig. 1. That specific rule would transform `malinha` (`little briefcase`) into `mala`, by replacing the affix `inha` for the affix `a`, but would leave `farinha` untouched, as it is one of that rule’s exceptions.

Although the rules for all types of normalization shared the same general structure, the rules regarding verbs were somewhat specific. (i) The rules for lemmatizing irregular verbs consisted of all the possible conjugations of the Portuguese irregular verbs. It was simpler to do this than to come up with rules that could address all existing variations. (ii) The rules for the regular verbs used as target the stem of each verb, always ending in a consonant, followed by the only vocals that could be appended to that specific stem (depending on the conjugation the verb belongs to), ending with any sequence of letters. Small variations that could occur – for instance, the substitution of a `g` for a `j` in the verb `agir` (`to act`) in some of its inflections – were also considered. (iii) When the two previous types of rules failed to be applied, a set of rules with verbal inflection affixes was used.

For this to be possible, regular expressions were used. Also, any of these rules could accept a list of prefixes, to broaden the list of addressed verbs. An example of rules for regular verbs is shown in Fig. 2, where it is also shown the list of prefixes that can be added to a verb, and the ending (suffix) of all the verb rules.

Eventually, in the selection of the rule that would be applied to a word in a given step, beyond its target and syntactic category, when more than one rule was eligible, the lengthier one was chosen. The length of a rule was computed by a weighted sum of the number of characters in the target, the exceptions and the POS tags, from a higher to a lower weight.

It is worth noticing that the rules are easily readable and customizable. Moreover, it is possible to select which kind of normalization steps should be performed, by specifying flags on the calls to the lemmatizer – when none is specified, it defaults to apply all the normalization steps. Both of these features make our lemmatizer flexible and easy to adapt to different situations and purposes.

3.2 The Addition of a Lexicon

The current version builds up on the previous (using rules), with the addition of a lexicon, namely the “LABEL-LEX-sw” lexicon,[5] version 4.1, produced by LabEL [7]. The specified lexicon contains over 1,500,000 inflected forms, automatically generated from about 120,000 lemmas, characterized by morphological and categorical attributes.

[5] The LABEL-LEX-sw lexicon is provided by LabEL, through `http://label.ist.utl.pt`.

```
gata,gato.N+z1:fs
gatas,gato.N+z1:fp
gatinha,gato.N+z1:Dfs
gatinhas,gato.N+z1:Dfp
gatinho,gato.N+z1:Dms
gatinhos,gato.N+z1:Dmp
gatita,gato.N+z1:Dfs
gatitas,gato.N+z1:Dfp
gatito,gato.N+z1:Dms
gatitos,gato.N+z1:Dmp
gato,gato.N+z1:ms
gatos,gato.N+z1:mp
```

Figure 3 An example of "LABEL-LEX-sw" lexicon entries.

Beyond using this lexicon for validating the lemmas produced by the lemmatizer, we have used the fact that each entry of the lexicon contains the inflected form, lemma, syntactic category, syntactic subcategory, and morphological attributes, that can be directly applied in the lemmatization process. Fig. 3 shows an example of these entries.

Using this lexicon provided an easy way of retrieving the lemma of any word, given its syntactic category. Also, the rules previously defined are now used only when a word is not found in the lexicon, with one advantage: virtually all exceptions to the rules are already present in the lexicon, so that the probability of a rule failing is extremely low. This comes from the fact that the exceptions are usually found in extremely frequent, ancient, and well known words of a lexicon, rather than in more recent, less used, or obscure words.

However, this does not mean that the lexicon could be used right out of the box. Some issues had to be addressed: (i) a mapping between the syntactic categories present in the lexicon and the ones used on the rest of the program (including the rules used in earlier versions); (ii) excluding all pronoun and determiner lexicon entries, as they present disputable normalization – for instance, `tu` (`you`) has `eu` (`I`) as its lemma; and (iii) making optional some gender normalizations, such as presenting `homem` (`man`) as the lemma of `mulher` (`woman`).

When a word is shared by multiple lemmas, the lemma with the highest frequency is selected. For this purpose, we used the frequency list of the combined lemmas present in all the Portuguese corpora available through the AC/DC project [11].[6]

The only drawback of the method is the time it takes, even if only a couple of seconds, to load the lexicon into memory. Other than that, it is quite performant, as the lexicon is stored in a hash structure, that is known to be a fast method for storing and searching on sets of elements. A lemma cache is also used, with each word that is found in the analysed text to be stored together with is syntactic category (POS tag) and lemma, at run-time, which avoids searching again the whole lexicon or selecting which rule to apply for a word already processed. The cache is a particular improvement to performance speed because, besides a set of words commonly used across different domains, texts on a specific topic tend to have their own set of words that are used multiple times over and over again. The basic structure of the currently used lemmatization algorithm is presented in Listing 1.

Regarding flexibilization, besides the customization of rules and selection of which normalization steps to apply, the current version of the lemmatizer allows the option to add new entries to a custom lexicon (if it fits best to do so in a lexicon, instead of specifying an exception in the rules, or both).

[6] The frequency lists of AC/DC are provided by Linguateca, through `http://www.linguateca.pt/ACDC`.

Listing 1 Overview of the lemmatization algorithm used.

```
load lexicon;
load rules;

lemmatize (token, tag) {
  if cache contains (token, tag) {
    return lemma of (token, tag);
  }
  if lexicon contains (token, tag) {
    add (token, tag) to cache;
    return lemma of (token, tag);
  }
  lemma = token;
  for each rule in (adverb, number, superlative, augmentative,
      diminutive, gender, verb) {
    lemma = normalize (lemma, tag, rule);
    if lexicon contains (lemma, tag) {
      add (token, tag) to cache;
      return lemma of (token, tag);
    }
  }
  return lemma;
}
```

4 Evaluation and Results

For the lemmatizer evaluation, we have used Bosque 8.0, the last version of a manually revised part of the Floresta Sintática treebank [1], by Linguateca.[7] Bosque contains around 120,000 tokens with annotations at various syntactic levels, for the Portuguese portion, and around 70,000 for the Brazilian portion.

Bosque was parsed in order to retrieve, for each word found in it, the inflected form, its syntactic category and corresponding lemma. The inflected form and syntactic category were fed to our lemmatizer, and the output was matched against the known lemma, as identified in Bosque.

In Table 1 we can see the overall results using rules, the lexicon, and both rules and lexicon (the current version of the lemmatizer), applied to the Portuguese and Brazilian parts of Bosque, with the current version reaching an accuracy value over 98%. The same table also presents the results broken down into three major syntactic categories: nouns, adjectives, and verbs.

It is possible to notice that using the lexicon greatly improves the normalization of adjectives (although other categories also benefit from it) against using only rules. However, the lexicon only by itself does not cover all the cases either, as it is virtually impossible for a lexicon, comprehensive as it may be, to cover all the lexemes, and associated syntactic categories, in any language. For instance, past participles in the plural form are not contemplated in the used lexicon – that may be one of the reasons rules perform better than the lexicon on verbs, beyond having a more extensive verb list.

[7] Floresta Sintática is freely available from `http://www.linguateca.pt/floresta/BibliaFlorestal/completa.html`.

Table 1 Overall and partial results in major categories.

Bosque	Only Rules	Only Lexicon	Both Rules and Lexicon
Overall PT	97.76%	95.06%	98.62%
Overall BR	97.67%	95.16%	98.56%
Nouns PT	96.94%	98.05%	98.30%
Nouns BR	96.40%	96.67%	97.86%
Adjectives PT	90.10%	95.39%	98.19%
Adjectives BR	88.77%	91.70%	97.23%
Verbs PT	98.04%	88.78%	98.79%
Verbs BR	98.34%	89.59%	99.15%

Table 2 Errors and discrepancies identified in both the lemmatizer and Bosque.

Type	Quantity	Example (*Form#POS:Bosque:LemPORT*)
Incorrect categorization	1.25%	`Afeganistão#N:afeganistão:afeganisto*`
Orthographic errors	3.75%	`ecxemplares*#N:exemplar:ecxemplar*`
Both lemmas acceptable	2.50%	`cabine#N:cabine:cabina`
LemPORT errors	43.75%	`presas#V-PCP:prender:presar`
Bosque errors	48.75%	`pais#N:pais:pai`

The accuracy could be even higher (probably slightly above 99%), as in a significant amount of the faulty cases the problem may actually be found in the Bosque annotation. In other discrepancies, different lemmas could be accepted, depending on the purpose. A brief analysis of a random sample of 10% (160) of the cases where the lemmatizer produced different lemmas on the Portuguese part of Bosque is presented in Table 2.

5 Conclusions and Future Work

We have presented a cross-platform lemmatization tool for Portuguese developed in Java that, through the use of simple rules in conjugation with a comprehensive lexicon, is able to have a very high overall accuracy, over 98%, making it suitable for use in many NLP tasks. For instance, its earlier versions have been used in a question generation system [6], and in the creation of the lexical-semantic resources CARTÃO [8] and Onto.PT [9]. We are also currently using this lemmatizer in a question-answering system under development.

Although the margin for improvement is narrow, we still hope to improve the lemmatizer by addressing some minor but troublesome issues, such as composed and hyphenated words, as well as multiword expressions. We already partially tackle one of these issues, by splitting the words at the hyphen, sharing the syntactic function. However, there are cases where elements of composed and hyphenated words, when put apart, belong to different categories.

Other issues may include the processing of oblique cases in pronouns. Bosque presents the oblique case and the pronoun that would be the corresponding lemma, but we usually process the oblique cases prior in the tokenization process.

We also intend, in the near future, to compare the lemmatizer against jSpell and Freeling, using the Bosque data as input, and processing the output of both tools in order to extract only the lemmas.

In order to be used by other members of the community, the presented lemmatizer is freely available from `https://github.com/rikarudo/LemPORT`, under the moniker "**LemPORT**".

Acknowledgements. This work was supported by the iCIS project (CENTRO-07-ST24-FEDER-002003), co-financed by QREN, in the scope of the Mais Centro Program and European Union's FEDER.

References

1 Susana Afonso, Eckhard Bick, Renato Haber, and Diana Santos. "Floresta Sintá(c)tica": a Treebank for Portuguese. In Manuel González Rodríguez and Carmen Paz Suárez Araujo, editors, *Proceedings of LREC 2002, the Third International Conference on Language Resources and Evaluation*, pages 1698–1703, Paris, 2002. ELRA.

2 Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval.* ACM Press, New York, USA, 1999.

3 António Branco and João Silva. A Suite of Shallow Processing Tools for Portuguese: LX-Suite. In *EACL'06 Proceedings of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics: Posters & Demonstrations*, pages 179–182, Trento, Italy, 2006.

4 António Branco and João Silva. Very High Accuracy Rule-Based Nominal Lemmatization with a Minimal Lexicon. In *XXII Encontro Nacional da Associação Portuguesa de Linguística*, pages 169–181, 2007.

5 Xavier Carreras, Isaac Chao, Lluís Padró, and Muntsa Padró. FreeLing: An Open-Source Suite of Language Analyzers. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC'04)*, pages 239–242, 2004.

6 Daniel Diéguez, Ricardo Rodrigues, and Paulo Gomes. Using CBR for Portuguese Question Generation. In *Proceedings of the 15th Portuguese Conference on Artificial Intelligence*, EPIA 2011, pages 328–341, Lisbon, Portugal, October 2011. APPIA.

7 Samuel Eleutério, Elisabete Marques Ranchhod, Cristina Mota, and Paula Carvalho. Dicionários Electrónicos do Português. Características e Aplicações. In *Actas del VIII Simposio Internacional de Comunicación Social*, pages 636–642, 2003.

8 Hugo Gonçalo Oliveira, Leticia Antón Pérez, Hernâni Costa, and Paulo Gomes. Uma Rede Léxico-Semântica de Grandes Dimensões para o Português, Extraída a partir de Dicionários Electrónicos. *Linguamática*, 3(2):23–38, December 2011.

9 Hugo Gonçalo Oliveira and Paulo Gomes. ECO and Onto.PT: A Flexible Approach for Creating a Portuguese Wordnet Automatically. *Language Resources and Evaluation*, to be published (online September 2013), 2013.

10 Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition.* Prentice Hall series in artificial intelligence. Prentice Hall, Pearson Education International, Englewood Cliffs, NJ, 2nd edition, 2009.

11 Diana Santos and Eckhard Bick. Providing Internet Access to Portuguese Corpora: the AC/DC Project. In *Proceedings of 2nd International Conference on Language Resources and Evaluation*, LREC 2000, pages 205–210, 2000.

12 Alberto Manuel Simões and José João Almeida. jSpell.pm – Um Módulo de Análise Morfológica para Uso em Processamento de Linguagem Natural. In *Actas da Associação Portuguesa de Linguística*, pages 485–495, 2001.