TÉCNICO LISBOA

# Fine-grained POS-tagging

Full disambiguation of verbal morpho-syntactic tags

**Joana Catarina Lapas Pinto**

Thesis to obtain the Master of Science Degree in

## Information Systems and Computer Engineering

Supervisors: Professor Doctor Nuno João Neves Mamede
Professor Doctor Jorge Manuel Evangelista Baptista

### Examination Committee

Chairperson: Professor Doctor Alberto Manuel Rodrigues da Silva
Supervisor: Professor Doctor Nuno João Neves Mamede
Member of the Committee: Professor Doctor Ricardo Daniel Santos Faro Marques Ribeiro

**June 2016**

# Acknowledgments

First, I would like to thank both my advisors, Prof. Nuno Mamede and Prof. Jorge Baptista, for their guidance, patience and for always supporting me through this course, with their optimism, wise advices and expertise, which made this work possible.

I would also to thank my parents for supporting my choices and giving me the opportunity to pursue a higher education. I am also thankful to my brother and my sister for their encouragement.

Last, but definitely not least, I would like to thank to my closest friends for their patience, optimism and for always believing in me.

# Abstract

Part-of-speech (POS) tagging is an important Natural Language Processing task and many systems have been applied to this problem, adopting either a rule-based, a probabilistic or a hybrid approach. However, most of the standard POS taggers do not disambiguate fine-grained morphological information within word categories. This information, such as gender and number, is important in parsing highly inflectional languages. European Portuguese presents a complex verbal inflection system, with many inflected ambiguous verb forms. This work aims at disambiguating verb forms, considering the inflection features of mood, tense, person, number and gender. To solve this problem, Machine Learning (ML) techniques were implemented in the STRING system. These ML techniques were tested in several scenarios, in order to analyse the impact of all the possible sequences of inflection features on performing the full disambiguation of the verb tag. Among the tested ML methods, Maximum Entropy (ME) obtained the highest precision (95.28%) in the disambiguation of verbal inflection features, contrasting with the baseline that only reached 91.67%. This baseline is a result of verbal inflection disambiguation rules combined with statistical disambiguators for the disambiguation of the category and subcategory, verbal lemma, along with the disambiguation of personal pronouns.

# Keywords

Natural Language Processing; Machine Learning; Part-of-Speech Tagging; Verbal Inflection Disambiguation; Inflection Features; Verbal Lemma Disambiguation

# Resumo

A anotação morfossintática constitui uma das principais tarefas no Processamento de Língua Natural, pelo que existem vários sistemas que têm abordado este problema, que se têm vindo a basear no desenvolvimento de regras, métodos estatísticos ou abordagens híbridas. Ainda assim, muitos destes sistemas não são desenvolvidos a ponto de desambiguarem totalmente a informação morfológica de uma palavra. Esta informação é importante na análise sintática em línguas com um sistema flexional mais complexo. A Língua Portuguesa tem um sistema morfológico complexo na flexão verbal, apresentando muitas formas verbais ambíguas na sua flexão. O objectivo deste trabalho é a desambiguação de formas verbais, considerando a flexão em modo, tempo, pessoa, número e género. Para resolver este problema, foram implementados métodos de Aprendizagem Automática no sistema STRING. Estas abordagens foram testadas sob diferentes condições, de modo a fazer uma análise do impacto observável quando se faz variar a ordem pela qual as características de flexão são desambiguadas, a fim de a desambiguar a totalidade da etiqueta morfosintática de uma forma verbal. O método da Máxima Entropia foi o que atingiu um valor mais alto de precisão (95,28%) na desambiguação das características de flexão. Este valor contrasta com o limiar definido, que se situou num valor de 91,67% para a precisão. Este limiar baseia-se numa combinação de um sistema de regras, que inclui situações específicas na desambiguação da flexão das formas verbais, com desambiguadores estatísticos construídos para a desambiguação de categoria e subcategoria de uma palavra, desambiguação do lema verbal e desambiguação de pronomes.

# Palavras-Chave

Processamento de Língua Natural; Aprendizagem Automática; Anotação Morfosintáctica; Desambiguação da Flexão Verbal; Características de Flexão; Desambiguação do Lema Verbal

# Contents

# List of Figures

# List of Tables

# Acronyms

| | |
|---|---|
| **NLP** | Natural Language Processing |
| **POS** | Part-of-speech |
| **HMM** | Hidden Markov Model |
| **WSD** | Word Sense Disambiguation |
| **SVM** | Support Vector Machine |
| **ME** | Maximum Entropy |
| **ML** | Machine Learning |

# 1

# Introduction

## Contents

People communicate with each other through language, either orally or in writing. Thus, why not communicate with a machine in the same way? This form of communication makes the human-computer interaction more natural to the user. However, this process brings complexity to the machine, that has to support Natural Language Processing (NLP). This comprises text segmentation, Part-of-speech (POS) tagging, morpho-syntactic disambiguation, shallow parsing, and deep parsing, among others.

Text-to-Speech (TTS) systems are an example of human-computer interaction, where the computer outputs a spoken voice from a text input. The more information is obtained from the text, the better is the speech the systems produce [22]. For instance, knowing the POS tag for a given word is useful to pronounce that word correctly. For example, the Portuguese word *almoço* 'lunch' can be classified both as a noun (and it is pronounced as [aL'mosu] (closed "o") and as a verb (then with the pronunciation [aL'mOsu] (opened "o")).

There are other applications where it is relevant to know which POS tag is assigned to a word. An example of these applications is the development of language models for speech recognition.

As previously mentioned, some words have more than one possible tag (e.g. *almoço* 'lunch') and, as such, they constitute a problem, since the most adequate tag must be chosen. A POS tagger is a system which assigns a POS tag to each word. These systems have a fixed set of tags to assign, known as the *tagset*. It is important to note that the size of the tagset varies from system to system, as some systems consider just the main categories, like nouns, verbs, and adjectives, while others systems have a more fine-grained tagset, with inflection features, such as number, person, gender, tense and mood for the verbs.

This project aims at disambiguating the European Portuguese inflection features of verbs, as well as disambiguating the lemma of some particular verb forms, a problem that will be detailed in Section 1.2. The work proposed in this document will be integrated into MARv4, which is a module of the STRING system, which will be presented next. Figure 1.1 shows the modules of the STRING chain.

## 1.1 STRING

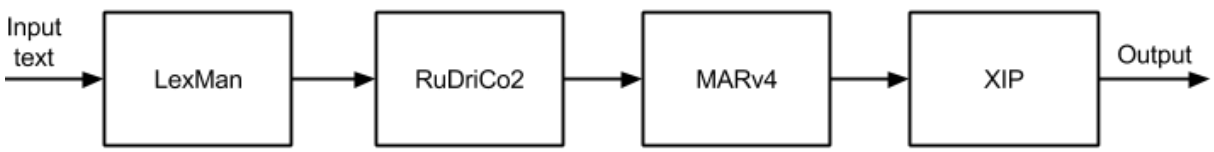


**Figure 1.1:** Main Modules of the STRING chain.

STRING[1] is a hybrid statistical and rule-based NLP chain for Portuguese [18]. It has been developed by L2F-Spoken Language Laboratory at INESC-ID Lisboa since 2006. Nowadays, STRING is a powerful system managed to perform several NLP tasks, such as Named Entity Recognition, Information Retrieval and Anaphora Resolution [18]. The NLP chain is organized in four modules (see Figure 1.1) in order to perform the basic text processing tasks: text segmentation (text tokenization and sentence splitting), part-of-speech tagging, morphosyntactic disambiguation, shallow parsing (chunking) and deep parsing (dependency extraction) [18].

The first step of text processing is text segmentation and tokenization. This task consists in the splitting of the text into sentences and the delimitation of the text's linguistic units (tokens) that are to be tagged next with all relevant linguistic information [18]. Within STRING system, LexMan [32] is the module responsible for this task. It is a lexical analyser and besides text segmentation, it also assigns to each segment all possible POS tags and any other relevant inflection features, using finite-state transducers. LexMan uses a tagset featuring 12 categories (noun, verb, adjective, pronoun, article, adverb, preposition, conjunction, numeral, interjection, punctuation, and symbol) and 11 fields (category (CAT), subcategory (SCT), mood (MOD), tense (TEN), person (PER), number (NUM), gender (GEN), degree (DEG), case (CAS), syntactic features (SYN), and semantic features (SEM)) [18]. A complex mapping defines which fields and their respective values can be associated to each category (see Appendix A).

The system then proceeds with two distinct morphological disambiguator modules. The first one is RuDriCo2 [10], a rule-based morphological disambiguator, which comprises segmentation and disambiguation rules. RuDriCo2 introduces several improvements in the results produced by LexMan, eventually modifying the initial text segmentation. By way of segmentation rules, RuDrico2 is able to develop either contractions into their constituent elements or capture in a single token several segments (multi-word units). This module expands, for example, the expression *nas* into *em* and *as*. On the other hand, all the contractions are context-dependent, for instance, given the segments *cerca* and *de* followed by an cardinal number such as *50*, this module would aggregate the segments *cerca* and *de* into a single token, *cerca de* 'about'. Another capability of RuDriCo2 is POS disambiguation, assigning the most adequate POS to a segment, using hand-written disambiguation rules, which match surrounding words that define their current word context.

MARv4 [21] is the other morphological disambiguator and it is the following module in the processing chain. It is a statistical disambiguator, supported by HMM, that chooses the most likely POS tag for each token. It uses second-order language models, using tri-grams to codify contextual information concerning entities and lexical information codified in unigrams. In this way, MARv4 disambiguates the tokens that RuDriCo2 did not solve. To do so, MARv4 uses the Viterbi algorithm to select the best POS tag for each token given its previous context. Furthermore, over the past few years, this

[1] https://string.l2f.inesc-id.pt/; a demo is also available at: https://string.l2f.inesc-id.pt/demo/ [2016-06-03]

module has been upgraded and its current version is also able to disambiguate verbal lemmas. For instance, for the verb form *foi*, both lemmas *ser* “be” and *ir* “go” are possible. Additionally, MARv4 deals with personal pronouns such as the ambiguity between reflex, dative, and accusative pronouns. The ambiguity between nominative (subject) and oblique (prepositional object) is also solved by this module.

XIP [1], the next module of the chain, performs syntactic analysis producing a chunking tree and determining the syntactic dependencies between words and chunks. Chunking rules are used to group sequences of categories into larger structures, for example, grouping an article and a noun into a Noun Phrase (*NP*). XIP also has dependency rules that take the sequences of constituent nodes identified by chunking rules and identifies relationships between them. XIP is also used to add syntactic and semantic features to the lexicon, which are necessary for an adequate parsing. This information is usually added to the lemmas. It is therefore very important that lemmas and other morpho-syntactic features be correctly disambiguated at different levels of granularity so that the parser is able to adequately produce its analysis.

## 1.2 Problem

As other Romance Languages, one of the most prominent features of the Portuguese language is the complexity of its verbal inflection system [5]. The processing of verb forms deals with two types of ambiguities: the first one concerns to the lemma, while the second one concerns the inflection features such as tense (temporal, aspectual and modality values), number or gender for past participles. The verb form *aposto* is a example of the first type of ambiguity, as it have two distinct lemmas, *apostar* ‘bet’ and *apor* ‘add’.

In order to disambiguate verb forms, it is necessary to analyse the context of that word. However, in some situations, lemma disambiguation is not enough to disambiguate a verb form completely, i.e. assigning the right lemma and all its inflection features. For instance, to relate events, in some cases, it is essential to know the most suitable tense of a verb form to establish a temporal relation between events. For example, in the clause *O Rui já jantara quando o João chegou a casa* ‘Rui had already had dinner when João arrived home’, the main clause is in Pluperfect past tense and the temporal subordinate clauses is in Perfective past tense. By the use of the subordinate conjunction *quando* ‘when’ in the sentence, and the tenses it is possible to ensure that the second event (subordinated clause) occurred before the first event (main clause). In addition of tense disambiguation, there are other inflection features that may have to be clarified. Next, Table 1.1 shows the verb form *acaba* with several possible inflection feature values, specifically to the mood, tense and person.

There are also some verb forms where these ambiguity types may be combined, i.e., with several possible combinations of lemmas and inflection features. As exemplified in Table 1.2, there are two pos-

| Verb form | Lemma | Mood | Tense | Person | Number | Gender |
|---|---|---|---|---|---|---|
| *acaba* | *acabar* | Indicative | Present | $3^{rd}$ | Singular | Undefined |
| *acaba* | *acabar* | Imperative | Undefined | $2^{nd}$ | Singular | Undefined |

**Table 1.1:** Possible lemmas and inflection features for verb form *acaba*.

| Verb form | Lemma | Mood | Tense | Person | Number | Gender |
|---|---|---|---|---|---|---|
| *virmos* | *ver* | Subjunctive | Future | $1^{st}$ | Plural | Undefined |
| *virmos* | *vir* | Inflected infinitive | Undefined | $1^{st}$ | Plural | Undefined |

**Table 1.2:** Ambiguity in lemma and inflection features of verb form *virmos*.

sible lemmas for the verb form *virmos* and two possible values for mood (modality) and tense features. In this case, as each lemma has different features values, a correct choice of lemma also represents an adequate assignment of mood and tense for the verb form. However, for some verb forms, the choice of correct lemma can not solve entirely the problem of disambiguating inflection features. The verb form *foram*, as exemplified in Table 1.3, if the correct lemma is *ser* 'be', there are still two possible values for the tense feature. Therefore, in order to make a system able to make a full disambiguation of the verbal tag, there is a need for another mechanism besides just lemma disambiguation.

### 1.2.1 Facts and Figures in Portuguese language

In this subsection, an overall analysis of the information on verb forms in STRING will be presented, in order to have a better understanding of the real impact of the lemma and inflection-derived ambiguity in European Portuguese.

As it was previously mentioned, LexMan [32] is the module responsible for generating all the possible verb forms in Portuguese. To achieve this, it has a submodule, LexManGenerator. This submodule has two types of input files: lemma files and paradigm files. In lemma files, each lemma is associated with a stem that is used to generate inflected forms. Additionally, there is also a corresponding inflectional paradigm, which is applied to the lemma stem in order to generate all possible inflected verb forms associated to that lemma. Paradigm files contain the transformations that have to be performed in the lemma stem so that the inflected verb forms may be produced. Besides that, a tag is also assigned to each verb form, explicitly encoding all the linguistically relevant morpho-syntactic features (inflection

| Verb form | Lemma | Mood | Tense | Person | Number | Gender |
|---|---|---|---|---|---|---|
| *foram* | *ir* | Indicative | Pluperfect past | $3^{rd}$ | Plural | Undefined |
| *foram* | *ser* | Indicative | Perfective past | $3^{rd}$ | Plural | Undefined |
| *foram* | *ser* | Indicative | Pluperfect past | $3^{rd}$ | Plural | Undefined |

**Table 1.3:** Ambiguity in lemma and inflection features of verb form *foram*.

values, mostly) annotated to that verb form. Finally, with this input, LexManGenerator produces a file containing all the inflected verb forms and each one of them is associated with a lemma and a tag.

LexMan processes 11,496 different verb lemmas and, combined with inflectional paradigm it generates 585,320 inflected verb forms, which represent all the verbs conjugated possible combinations of the categories mood, tense, person, number, and gender, with 80 distinct tags for verbs, within a total of 375 tags approximately. It is important to note that this number does not include verb forms with *clitics*. A *clitic* is a morpheme that behaves syntactically as a word but is phonologically dependent on another word or phrase. Depending on the position they hold in relation to the verb *clitics* are classified as *proclitic*, if *clitic* appears before the verb as in *não a viu* 'did not see her'; *mesoclitic*, when it appears in the middle of the verb *vê-la-ei* 'shall see her'; and *enclitic*, if it appears after the verb, *viu-a* 'saw her'. These verb forms are excluded in this analysis as they are not particularly relevant to the ambiguity quantification, since, in most cases, only one tag can be assigned to each word.

With respect to ambiguity in Portuguese, LexMan generates 137,266 distinct ambiguous verb forms. From this first set, 890 are related with lemma ambiguity. On the other hand, 137,253 verb forms are ambiguous in terms of their inflection features, in a total of 308,748 ambiguous tags plus form cases, according to the equation 1.1, which means that the total number of ambiguity cases is the sum of the number of verb forms with $n$ distinct tags, represented by $|vfntags|$ multiplied by the $n$; the number of tags varies between two and five;

$$TotalOfAmbiguities = \sum_{n=2}^{5} n|vfntags| \tag{1.1}$$

From the total of verb forms with ambiguous inflection features, approximately 81.4% have five tags, 18.2% have two tags, 0.3% have three tags and 0.1% have four tags. Besides that, there are also approximately 734 verb forms with both ambiguities (lemmas and inflection features).

Since approximately 99.99% of ambiguous verb forms concern ambiguity on inflection features, the analysis and disambiguation of the most adequate set of inflection features to a certain verb form is a relevant work on NLP.

MARv4 is already able to decide, out of the possible lemmas, which is the most appropriate for 36 verb forms, which represent approximately 4% of the total lemma-ambiguous verb forms. These verb forms will be discussed later in Chapter 3.

## 1.3 Goals

The main goals of this project are:

- Disambiguate the full tag of a given word, after it has been classified as a verb;

- Analyse and compare the better performing strategy to disambiguate ambiguous verb forms in lemma and inflection features: whether to disambiguate first the lemma or the inflection features.
- Improve disambiguation of verbal lemmas;
- Improve disambiguation of case of personal pronouns;
- Integrate the developed module on STRING, introduced in Section 1.1.

# 2

# Related Work

## Contents

## 2.1 Verbal Morphological Disambiguation

In this subsection, two approaches will be presented in order to analyse different solutions for disambiguating ambiguous verb forms, with respect to their inflection features. These systems are language-dependent: the first one is applied to Portuguese and it is based on HMM; and the second one was built for German, based on manually-crafted rules.

In 2012, Branco *et* al. [6] presented the task of verbal inflection analysis as a task similar to POS tagging, for European Portuguese. To build a verb analyser, TnT [7] was used, implementing a HMM approach with back-off and suffix analysis. The model was trained with a fraction of the CINTIL *corpus* [3], comprising 261,385 tokens, containing parts from news (3/5) and from novels (2/5). This is a manually annotated *corpus* with a large tagset, including a subset of 80 tags (bundles of feature values) for verbal inflection in Portuguese. 90% of the data was used to train the model, where are included 27,823 verb forms and the remaining 10% was used to evaluate it, with 3,153 verb forms. Additionally, to compare the performance of the verb analyser with a POS-only tagger, the TnT was trained and tested under the same conditions of verb analyser, considering a basic annotation with 69 tags, to obtain a POS tagger. The produced tagger has achieved 96.87% of precision. Lastly, several inflection disambiguators were produced, although no details were provided about how disambiguators were implemented.

Reasonably, several experiments were made to obtain the optimal parameters that achieve the best precision for the verb analyser. Firstly, the experiments will be presented, and then the results performed by each approach will be discussed, emphasizing precision values.

In the first experiment, the HMM was inputted with accurate, manually-annotated, POS tags and its output was the verb inflection tags, in case of verb tokens, or null, in the case of the remaining tokens. In the second experiment, the POS tags were assigned automatically (by the POS tagger referred above), and then the verbal analyser was tested under these circumstances. In the last experiment, the input used was raw text and the POS tagger and the verb analyser were trained as a single classifier. A larger tagset with morphological information was also included. The HMM then outputted the POS tag of a plain word, concatenated with the respective inflectional tag. Results are presented in Table 2.1.

According to Table 2.1, the first classifier reached the best performance, with a precision of 93.34%. Comparing this result with the second experiment, the first one is slightly better, due to a 97.0% of precision of the used POS tagger. Therefore, in the second experiment, the verb analyser was run over data that was more likely to contain misplaced POS tags, corresponding to verbs. From the experiment 2 to the experiment 3, in spite of the effort to improve the solution, its precision slightly decreased. It can be concluded that the larger tagset provided no benefits due to the sparseness of data.

The problem of analysing verb inflection can be seen through another perspective, in a linguistically informed way. Under this perspective, our problem can be conceptualized as a Word Sense Disam-

| Input | Output | Precision |
|---|---|---|
| Accurate POS | Infl tags | 93.34% |
| Automatic POS | Infl tags | 92.22% |
| Raw text | POS+Infl | 92.06% |

**Table 2.1:** Precision of HMM-based classifiers used for verb inflection analysis.

biguation (WSD) task. Therefore, to determine the most adequate inflection features of a verb form, the context in which a word occurs must be analysed.

In the same work, Branco *et* al. [6] presented an algorithm to solve the problem, with a heuristic-based method. A heuristic-based method is a subclass of knowledge-based methods from the WSD group. There are some known heuristics, such as the selection of the most frequent sense, methods based on lexical chains: one sense per discourse and one sense per collocation [6]. The algorithm was named the Most Frequent Feature Bundle (MFF) and it considers:

- TD as the training data;
- ST as the verbal inflection tags occurring in the training data;
- VF as the target verb form;
- AT as the set of admissible inflection tags.

With the elements above mentioned, the algorithm proceeds as follows:

1. If VF was observed in TD, from the tags $T_1$...$T_n$ in AT, pick $T_k$ such that VF_$T_k$ is more frequent in TD that any other VF_$T_n$;
2. Else if at least one tag $T_1$...$T_n$ in AT was observed in TD, pick $T_k$ such that $T_k$ is more frequent in TD than any other $T_n$;
3. Else pick a tag at random from AT.

Several variations of the verb analyser were tested, using the same training data as the verb analyser previously mentioned. In the first experiment, the analyser was run over a manually-annotated input. It achieved 96.92% precision, which was a better result comparing with the HMM analyser (see table 2.1) in the first experiment described above.

Then, the analyser was run over an input whose POS tags were automatically assigned by the POS tagger alone. This time, the analyser scored 94.73% in precision. As explained before, due to some incorrect POS tags, it was expected that this version of the analyser would present a worse precision score than the previous experiment.

In these statistical approaches, the problem of sparse data arises, since the analyser is trained with specific data. Therefore, as some words do not occur or occur just a few times in the training set, the next two experiments apply a smoothing technique. In order to obtain a better precision, if a verb form appears a number of times below a certain threshold, the analyser ignores it and proceeds according to step two of MFF algorithm. Table 2.2 groups the precision scores by all the thresholds the analyser has trained using the manually-annotated input. The best improvement was found with a threshold of 1 or 2 (96.98% of precision), which means that it is better to discard a verb form which appears only once in the training data. This version of the analyser improves precision by 0.04%, when comparing with the analyser under the same input conditions, but without smoothing.

| **Threshold** | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| **Precision** | 96.92% | 96.98% | 96.98% | 96.88% | 96.82% |

**Table 2.2:** Precision of verbal inflection analysis over input text with manually POS tags.

The next experiment used an automatically POS-tagged text. The results from this experiment are summarised in Table 2.3. Once again, smoothing could improve the analyser by 1.78%, when comparing the version without smoothing (threshold = 0) in Table 2.3.

| **Threshold** | 0 | 1 | 2 |
|---|---|---|---|
| **Precision** | 94.73% | 96.51% | 95.62% |

**Table 2.3:** Precision of verbal inflection analysis over input text automatically POS tagged.

Finally, Table 2.4 summarises a comparison of best results of HMM and MFF approaches. As it can be observed, the MFF analyser scores almost 4.3% points better than the best HMM analyser with an automatically POS-tagged input. To the accurately POS tagged data, MFF also presents a better result than the HMM-based approach.

| **Approach/ Input** | Accurate POS | Automatic POS |
|---|---|---|
| **HMM-based** | 93.34% | 92.22% |
| **MFF** | 96.98% | 96.51% |

**Table 2.4:** Comparison of best results.

Recently, in 2013, a work with the same purpose as that of [6] was presented by Sugisaki & Höfle [29], a rule-based system for German law texts, intended to perform verbal morpho-syntactic disambiguation through topological field recognition.

This work argued that POS tagging errors are reduced due to checking the compatibility of morpho-syntactic features, within long-distance syntactic relationships.

In the deployed system, if the rule-based system does not include the root of a word being analysed, a decision-tree-based system [26] is used to produce the morphological features. The work presents a German clause in terms of topological fields, which are the valid positions of non-verbal elements. Also, they are defined in terms of the positions of heads of the clause (e.g., finite verbs and their complementisers) and their verbal complements (e.g., infinitives, participles). Therefore, the verbal elements can be on the right or left sides of topological fields.

In the rule-based system, clauses are classified depending on the position of the verbal elements, which can be verb-first clauses (V1), verb-second clauses (V2) and verb-final clauses (VF). The first two clauses have in the left bracket a finite verb and on the right side have verbal complements. In the latter case, the content of left and right bracket are exchanged, when compared with V1 and V2.

The approach for verbal morpho-syntactic disambiguation has two main steps: in the first, it assigns the most adequate candidate tag to the verbal elements that occupy the left side, and identifies the clause type, by applying rules to the clauses; secondly, the verbal elements of the right side are disambiguated in terms of their morpho-syntactic features, applying rules and selecting the ones that are compatible with those which correspond to the left side elements. This is an incremental process as the tag disambiguation is done by the sequential application of the rules. In Figure 2.1, there is an example of rules to the left and right brackets.

**Left Bracket:**

```
The left bracket of V1 clauses is a single finite verb:  If a verb appears
in sentence-initial position, select the feature FINITE from its set of
possible features, mark it as left bracket and identify the clause type as V1.
```

**Right Bracket:**

```
A modal verb requires an infinitive:  If a potential infinitive is preceded
by a modal verb at the left-bracket position of a V1 or V2 clause, then
select its feature INFINITIVE and mark it as right bracket.
```

**Figure 2.1:** Rules to the left and right brackets of the rule-based system presented by Sugisaki & Höfle [29]

In order to evaluate the system, some sentences were extracted from the Swiss Legislation Corpus [15], where the left and right brackets were manually annotated. Then, 313 tokens from these sentences were used to test the rule-based system, which achieved a precision of 99.7%. This system has the advantage of analysing a wide context window since the left bracket and the right clauses are separated by approximately 9.5 tokens.

It is important to note that the two systems presented in this section cannot be directly compared since the data used to train and test the models are very distinct. Furthermore, the number of tags

used in the tagger of the German language is not explicit, and this has a high influence on the tagger's precision. Finally, the German system obtained a higher precision than the Portuguese (99.7% vs 96.58%), even though it was only tested with a few tokens.

## 2.2 POS Taggers

This section will present several approaches to the task of POS tagging: rule-based, probabilistic and hybrid approaches.

### 2.2.1 Rule-based Approaches

In the absence of annotated corpora, rule-based systems are an alternative approach to solve morpho-syntactic ambiguities.

Several approaches tried to implement POS tagging for Portuguese. In 2010, a system with four levels of rules was implemented by Seara *et al*. [27], to Brazilian Portuguese. These rules are based on the context surrounding each word, considering two or three items after or before the word being classified. To words that present more than one tag, there are rules that have to analyse the function of the word in a particular sentence.

The first of the four levels of the system is to pre-label the tokens with verbal root and verbal ending. Then, the second level labels multiword expressions, and the next one solves some ambiguities generated in the first level. Finally, the fourth level classifies sentences with WH questions.

It is extremely difficult for a human to remember all the classification rules, so this work also used an annotated corpus of medium-size, with 205,813 words and 25 tags to extract statistical information and create new rules. For words that are not matched by the rules, the system chooses the most frequent tag by default. Then, it compares manual annotations results with the first rules, in order to improve the precision of the system. Finally, it creates and refines new rules, taking into account the most frequent contexts. This is an incremental process, each time a new rule is created. The final classifier is composed of 649 rules and it is reported to have reached a precision rate of 99.0% for words and 82.03% for sentences.

### 2.2.2 Probabilistic Approaches

Probabilistic approaches are very common in POS tagging, such as HMM, which requires a large amount of training data to achieve high levels of precision. This section also presents some works using Support Vector Machine (SVM).

One of the first POS taggers for Portuguese was built in 1995 by Villavicencio [33]. For European Portuguese, 45 tags were considered, whereas, for Brazilian, only 35 tags were used. Nevertheless, inflection features were not included for any of them. The POS tagger used a first-order HMM, with bi-grams to model state transition probabilities. The tagger was trained and tested with RadioBrás *corpus* that contains news from the RadioBrás Agency from Brazil. This *corpus* contains 20,982 manually annotated words, including 267 ambiguous words. To overcome this problem, after the tags were assigned to the words, the Viterbi algorithm disambiguated words with more than one tag. This system achieved a precision of 87.98%.

A more recent work for Brazilian Portuguese, presented by Maia & Xexéo [9], opted by HMM, with a significant variation in the language model. Usually, a language model assigns a probability to a sequence of words but, in this particular case, it assigns a probability to a sequence of characters. Hence, emission probabilities are estimated using the relative frequent distribution of bounded character language models, one for each hidden state. This is an advantageous approach since it is able to include morpheme information and also estimate probabilities even for words which are not included in the training set. The Bosque *corpus* is a subset of Floresta Sintá(c)tica treebank [25] and it was used to train and test the model. The best results were achieved with 10-grams models and for a tagset with 39 tags, showing a precision of 96.2%. A more complex tagset was also considered, in order to include inflection information, covering 257 tags, which comprises gender, person, case, tense, and mood, as the tagset of STRING chain, described in the subsection 1.1. With this larger number of tags, the classifier reached a precision of 92.0%.

Considering other languages, in 1999, Tufis [31] proposed another probabilistic work to deal with large tagsets: a tiered, language-independent tagging system, organized in two levels. The first level used a condensed tagset, with 82 tags (plus 10 punctuation tags), and the second one used a more detailed tagset with lexical information.

The innovative idea here is that, from a tag assigned to a word of the reduced tagset, it is possible to recover the adequate morpho-syntactic descriptor in the large tagset. Hence, there is a function that maps a tag from the reduced tagset ($RT$) to a set of tags of the larger tagset ($LT$). This function is $MAP : RT \rightarrow LT_m$, where *m* is the number of tags. In 90% of the cases, this process is deterministic, so there is only a corresponding tag from RT to LT. In the remaining cases, there are 14 context rules to disambiguate the morpho-syntactic features of a word that inspect the surrounding context of that word, with a maximum span of 4 words.

Another concept raised in that work is the combination of classifying methods, wherein several classifiers are combined to reduce the number of errors. This process called error complementarity. However, in this work, a slightly different approach was proposed, as only one classifier is used, but tested with

several comparable-size corpora, in different registers, like fiction and journalism. For testing, 60,000 words were manually tagged and the classifier obtained a precision above 98.5%. Although the system is language-independent, the tests were done with Romanian texts.

Most of the referenced systems are implemented with supervised techniques but there are also experiments on POS tagging with purely unsupervised techniques, although they still achieve low precision, specially with multi-language systems. Thus, some works have appeared in order to improve such results, using weakly supervised learning, such as [14] and [17]. In 2012, Li *et al.* proposed a work [17] that used the Wiktionary, a freely available dictionary with large coverage of many languages, to deal with the problem of some languages having poor resources, and to overcome the costs of manually annotate corpora. This tool covers around 75 languages and, in this work, 8 of them were tested with a universal tagset of 12 categories. Several models were applied such as first and second-order HMMs (HMM and SHMM, respectively) and both with feature-based ME emission models (HMM-ME and SHMM-ME). The precision reported for the Portuguese language was 85.5% using SHMM-ME, and the best result was achieved for English (87.1%).

Over the years, some research on SVMs has emerged in the POS tagging task, such as [13] and [20]. These approaches are able to overcome the problem of some methods, like rule-based approaches and HMM, which have limitations on the information about the objects that are being considered, such as templates and the size of context for n-grams models, respectively.

One of the most important works in this area was proposed by Giménez & Màrquez in 2004. This is a tool based on Support Vector Machines [13]. This tool emerged in order to have a simple system, easy to configure and train, allowing for the capability of adjusting the size and shape of the feature context. Portability was also a requirement of this system since is intended as language-independent. Moreover, *linear kernels* were chosen due to the tagging time required efficiency. The SVM tool is composed by a learner, a tagger, and an evaluator. The first component is highly configurable and has a set of options: features can have different types, such as word forms n-grams, POS tags or affix and orthographic, such as hyphenization. The second component is also very flexible since it allows to choose the tagging scheme, either with a greedy approach, where the tag is assigned based on a reduced context or with at sentence-level based on dynamic programming, where the global sentence sum is considered for tagging. Another customizable option of the tagger is the tagging direction, which can be “left-to-right”, “right-to-left” or a combination of both. Furthermore, the tagger acts in two passes: in the first one considers only already disambiguated words; and in the second step, disambiguated POS features feed the words in the feature context. Experiments for English and Spanish were done. For English the SVM

used the Wall Street Journal *corpus* from the Penn Treebank, with 1,173,000 words, where 912,000 words compose the training set and 131,000 (11%) correspond to the test set. With this data, the English SVM achieved a precision of 97.16%. On the other hand, for Spanish the LEXESP[1] *corpus* was used with 106,000 words, where the training set had 86,000 words and the testing set had 20,000 (19%) words. This corpus has several sources, containing news items, press articles, and scientific texts. A precision of 98.86% was obtained with this tool.

Another work related to the previous machine learning algorithm was proposed in 2007 by Poel *et al*. [20]. It used a morpho-syntactically annotated *corpus*, Gesproken Nederlands [19], and a tagset with 316 different tags. A sliding window of seven words was adopted by the tagger, in order to tag the fourth word of the sequence, analysing the tokens left-to-right. The SVM POS tagger is decomposed in several taggers, in order to distribute the data by each one of them. The idea is that a word which occurs often in the corpus (more than 50 times) has its own multi-class SVM, since there are enough data [20]. If the word being tagged is uncommon, several SVM are trained, based on the tags of the previous word in the sequence. Several tests were done in order to choose the most suitable kernel to the SVM, wherein the polynomial of $3^{rd}$ order was the one with the best results. A precision of 97.52% was obtained, with a training set of 10,000 common words and 50,000 uncommon and unknown words.

The ME algorithm is also common in NLP, in tasks such as POS tagging. In MARv4, a ME algorithm is used to disambiguate the lemma of some particular verb forms and in the case of some personal pronouns, which will be enumerated in Chapter 3. In the training phase, it uses an external tool, MegaM [8], to build the models. Then, in the testing phase, the classifier loads the models in order to assign the most suitable class for a given instance. For all the potential classes of an instance, the tool computes its correspondent probability. Therefore, to determine the probability of each class $c$ given an instance $i$, represented by $P(c|i)$ the following formulas [4] are used:

$$P(c|i) = \frac{1}{Z(i)} \exp\left(\sum_k \lambda_k f_k(c,i)\right)$$

$$Z(i) = \sum_c \exp\left(\sum_k \lambda_k f_k(c,i)\right)$$

In the formulas above $f_k(c,i)$ is a feature of instance $i$ for class $c$ and $\lambda_k$ is the estimated weight in the model for that feature. To achieve the proper probability values the $Z(i)$ normalizing factor is used. When the probability values are computed, the classifier assigns the class which has the highest value to the instance.

[1] LexEsp *corpus* has been developed by the Computational Linguistics Laboratory at Barcelona University and the Natural Language Processing Group at UPC

### 2.2.3 Hybrid Approaches

Another technique that can be used in POS tagging is a combination of both rule-based and statistical approaches. Some examples of this hybrid approach will be presented below.

A project to build a grammar checker of Brazilian Portuguese in OpenOffice was carried out in 2003 by Kinoshita *et al*. [16]. To develop the system, a Brazilian Portuguese morpho-syntactic annotated *corpus* with 24 millions of words, named CETENFolha[2], was used. This project comprises a POS tagger that firstly assigns the most probable tag to a word, which is the one that appears more times in the corpus. Then, a more accurate approach is applied, based on contextual rules that assign a tag to a word depending on the tags of the three surrounding words (trigram). After the tagging task, rules based on patterns, which correspond to sequences of words or tags, are applied to detect errors as the defined article "a" before masculine names or adjectives. Then, if an error is detected, a message with suggestions for the correct sentences is shown to OpenOffice's user. To evaluate the system, the authors made a corpus with 16,536 manually annotated words. This work cannot be compared with other systems since the number of tags used in the POS tagger was not revealed. Nevertheless, a precision of 95.0% was reported.

While analysing other languages, another relevant work was proposed by Trushkina & Hinrichs [30], which applies a hybrid system to the highly ambiguous inflection of German language. This system is composed of three modules: a morphological analyser, a rule-based disambiguator, and a statistical tagger, instead of those systems with purely statistical disambiguation. The tagset used is based on the Stuttgart-Tubingnen tagset (STTS) and it was enriched with morpho-syntactic features, resulting in a tagset with 718 tags. The corpus used to train and test the model was the same, where 104 049 tokens were used in the first task and 11 361 tokens were assigned to the second one.

After the Xerox morphological analyser[3] performs the tagging task, the rule-based module proceeds with the disambiguation of these tags, which has two types of rules: agreement rules (like verbal and nominal agreement) and syntactic heuristics. Firstly, a POS disambiguation module is applied through the syntactic heuristic rules, which eliminate ungrammatical readings. This is done by applying sequential rules, containing constraints that the neighboring words impose on the word being tagged; for instance, a relative pronoun can be eliminated in the initial position of a sentence. Afterward, morphological ambiguities are solved over the two types of rules. This work enforces that agreement rules assume that, for instance, the lexical nodes of a verb phrase (VP) constrain each other in order to produce a valid set of possible interpretations. Finally, a statistical approach is applied, which is formed by a tagger

[2] http://www.linguateca.pt/CETENFolha/
[3] www.xrce.xerox.com/competencies/content-analysis/demos/german.de.html

based on probabilistic phrase structure grammars (PCFGs). These taggers can be a better approach since they are apt to include more global structural information when compared to n-gram models that consider the full surrounding context of the token being tagged. Therefore, the tagger decides what is the best sequence of tags, examining the maximal product of the probabilities of a tag, given a token, and the probability of a tag, given all the surrounding context. This statistical part was trained with 115,098 additional tokens from another corpus, using a weakly supervised approach, in order to reduce data sparseness.

It is important to note that the aim of the combined model is to take the advantages of both rule-based and statistical approaches, where the first one solves 70% of ambiguity tags and the second module solves the remaining tokens. The final model achieved a precision of 92.04%.

Table 2.5 shows all the results of the systems presented in this work, considering the precision values that they have obtained. Some relevant information about the corpora used in those works is included in the table, as the corpus sizes and the size of training and testing sets. These results cannot be compared directly since the corpora used by the systems is different and the number of tags assigned by POS taggers is also distinct and these two factors have a major influencing in the results. Nevertheless, for Portuguese, the first system [6] presented the highest precision, with 96.51%, maybe due to a good balance between the size of the corpus and the tagset, which includes inflection features. Moreover, as observed for the Brazilian Portuguese rule-based system [27], a precision of 82.02% was obtained when compared with other systems of the same language variety. The best system for Brazilian Portuguese was developed by [9], it is a probabilistic approach using HMM and it achieved a precision of 96.20%. For German, the best precision achieved was 99.7% and it is the best result of all the systems being composed here. However, it was tested only with a few tokens and the number of tags was not revealed nor the training *corpus* size. On the other hand, the other German system is the one with the largest tagset (718 tags) and obtained a precision of 92.04%. SVMs systems [13] and [20] also present good results, with a precision of 98.86% for Spanish, but the size of tagset was not revealed.

| Work | Language | Tagset size | Corpus size | Training set size | Testing set size | Precision |
|---|---|---|---|---|---|---|
| Branco et al. [6] | EP | 80 | 261 385 | 235 246 | 26 139 | 96.51% |
| Sugisaki & Höfle [29] | DE | - | 26 139 | - | 313 | 99.70% |
| Trushkina & Hinrichs [30] | DE | 718 | - | 219 147 | 11 361 | 92.04% |
| Maia & Xexéo [9] | BP | 39 | 180 000 | 162 000 | 18 000 | 96.20% |
| Kinoshita et al. [16] | BP | - | - | - | 16 536 | 95.0% |
| Maia & Xexéo [9] | BP | 257 | 180 000 | 162 000 | 18 000 | 92.60% |
| Villavicencio [33] | BP | 35 | 20 982 | 20 000 | 982 | 87.98% |
| Seara et al. [27] | BP | 25 | 205 813 | 174 941 | 30 872 | 82.03% |
| Tufis [31] | RO | 82 | - | - | 60 000 | 98.50% |
| Li et al. [17] | EN | 12 | - | - | - | 87.1% |
| Giménez & Màrquez [13] | ES | - | 106 000 | 86 000 | 20 000 | 98.86% |
| Poel et al. [20] | NL | - | - | 60 000 | - | 97.52% |

**Table 2.5:** Comparison of all the POS taggers presented.

# 3

# Corpora

## Contents

This chapter describes the training *corpus* used to choose the most likely category and subcategory for each token, a task already performed by MARv4. It also specifies the training *corpus* used to obtain the ML models applied to eliminate lemma ambiguity for verbs, to chose the case of personal pronouns and to disambiguate verb inflection features, which will be explained in detail in Chapter 4. Evaluation *corpus* is also presented at the end of this chapter.

## 3.1 Training *Corpus*

To assign the most adequate category and subcategory for each token and the remaining tag for ambiguous verb forms regarding inflection features, the training *corpus* is a segment of the LE-PAROLE [11], which has been manually annotated with STRING tagset.

The European project LE-PAROLE aimed to define and establish a collaborative infrastructure (CI) that would undertake the task of stimulating the creation and reuse of harmonized textual and lexical resources, as well as related tools throughout the community. For each language, 20 million words were collected, including 250 thousand morphosyntactically manually annotated words. Also, the lexicon of each language is composed of 20,000 entries, that contains syntactic and morpho-syntactic information. 20% of LE-PAROLE *corpus* was extracted from books, while 65% was extracted from newspapers; 5% from magazines and 10% from other resources (miscellaneous).

As LE-PAROLE is used to train the models for verbal inflection disambiguation, a brief analysis was made on the occurrences of verb forms in *corpus*, as well as their ambiguities.

The *corpus* contains a total of 38,927 verb forms, which corresponds to 15.6% of the total number of words. From the total of verbs, 19,220 (49.37%) have more than one admissible tag, concerning inflection features. It means that almost half of the verbs are an ambiguous case. This ambiguity was analysed and it was found that person inflection feature is the most ambiguous. There are 7,406 ambiguity instances in respect with person, which is the sum of each verb form with more than one value for person, multiplied by the number of admissible values for person of the mentioned verb form. Mood inflection feature was the second highest type of ambiguity, with 6,738 ambiguities. Additionally, there are 5,966 ambiguities for tense, 4,675 for number and, finally 3,505 for gender. Therefore, as these are significant numbers, the problem of disambiguation of verb forms has to be solved.

Table 3.1 represents ambiguous verb forms, regarding inflection features that occur more than 200 times in the *corpus* LE-PAROLE. The *ser* 'be' form is the one on the top of the list, with 655 occurrences, which is approximately 3.4% of the total of ambiguous inflected verb forms presented in *corpus*. This table also presents the tag of each verb form. Appendix A explains in detail the notation used in these tags.

| Verb form | # Occurences | Tags | | |
|---|---|---|---|---|
| *ser* | 655 | V.f=1s=..== | V.f=3s=..== | V.n====..== |
| *ter* | 438 | V.f=1s=..== | V.f=3s=..== | V.n====..== |
| *está* | 435 | V.ip3s=..== | V.m=2s=..== | |
| *tem* | 391 | V.ip3s=..== | V.m=2s=..== | |
| *era* | 389 | V.ii1s=..== | V.ii3s=..== | |
| *disse* | 322 | V.is1s=..== | V.is3s=..== | |
| *vai* | 298 | V.ip3s=..== | V.m=2s=..== | |
| *fazer* | 276 | V.f=1s=..== | V.f=3s=..== | V.n====..== |
| *tinha* | 234 | V.ii1s=..== | V.ii3s=..== | |
| *estava* | 231 | V.ii1s=..== | V.ii3s=..== | |
| *foram* | 209 | V.iq3p=..== | V.is3p=..== | |

**Table 3.1:** Ambiguous verb forms, with respect of inflection features, that occur in *corpus* LE-PAROLE more than 200 times and the admissible tags for each one.

| Verb form | # Occurrences | Lemmas | |
|---|---|---|---|
| *foi* | 855 | *ir* | *ser* |
| *foram* | 209 | *ir* | *ser* |
| *pode* | 196 | *podar* | *poder* |
| *seria* | 85 | *ser* | *seriar* |
| *fosse* | 84 | *fossar* | *ir* \| *ser* |
| *tendo* | 67 | *tender* | *ter* |
| *podem* | 61 | *podar* | *poder* |
| *vir* | 48 | *ver* | *vir* |
| *fora* | 39 | *ir* | *ser* |
| *for* | 34 | *ir* | *ser* |
| *visto* | 32 | *ver* | *vestir* |
| *fui* | 28 | *ir* | *ser* |
| *podes* | 26 | *podar* | *poder* |
| *podemos* | 22 | *podar* | *poder* |

**Table 3.2:** Verb forms with lemma ambiguity that occur in LE-PAROLE more than 20 times.

With respect to lemma ambiguity, there are 2,094 verb forms with more than one possible lemma, which is approximately 5.4% of the total verb occurrences in the *corpus*. Table 3.2 contains the verb forms which appears in LE-PAROLE more than 20 times, together with their lemmas. The verb *foi* is the most frequent, as it has 855 occurrences in the *corpus*, assuming lemma *ir* or *ser*. It is important to note that this verb form is one of the verbs which MARv4 is able to decide which lemma is more suitable, out of the possible lemmas. From the total of lemma ambiguity, there are 1,156 verb forms that have also several admissible tags, that is more than a half of the occurrences of verbs with lemma ambiguity. Table 3.3 shows these verb forms that occur more than 20 times in LE-PAROLE, as well as their acceptable lemmas / tags.

Regarding Portuguese personal pronouns, as previously mentioned in Section 1.1, MARv4 chooses the adequate value for case inflection feature for ambiguous pronouns. Case inflection can assume the values nominative, oblique, accusative, dative, and reflexive, as shown in Table 3.4.

| Verb form | # Occurrences | Lemmas / Tags |
|---|---|---|
| *foram* | 209 | *ir, ser* / V.iq3p=..== , V.is3p=..== |
| *pode* | 196 | *podar* / V.sp1s=..== , V.sp3s=..== |
| | | *poder* / V.ip3s=..== , V.m=2s=..== |
| *seria* | 85 | *ser* / V.c=1s=..== , V.c=3s=..== |
| | | *seria* / V.ip3s=..== , V.m=2s=..== |
| *fosse* | 84 | *fossar* / V.sp1s=..== , V.sp3s=..== |
| | | *ir*, *ser* / V.si1s=..== , V.si3s=..== |
| *tendo* | 67 | *tender* / V.ip1s=..== |
| | | ter / V.g====..== |
| *podem* | 61 | *podar* / V.sp3p=..== |
| | | *poder* / V.ip3p=..== |
| *vir* | 48 | *ver* / V.sf1s=..== , V.sf3s=..== |
| | | *vir* / V.f=1s=..== , V.f=3s=..== , V.n====..== |
| *fora* | 39 | *ir*, *ser* / V.iq1s=..== , V.iq3s=..== |
| *for* | 34 | *ir* , *ser* / V.sf1s=..== , V.sf3s=..== |
| *visto* | 32 | *ver* / V.p==sm..== |
| | | *vestir* / V.ip1s=..== |
| *podes* | 26 | *podar* / V.sp2s=..== |
| | | *poder* / V.ip2s=..== |
| *podemos* | 22 | *podar* / V.sp1p=..== |
| | | *poder* / V.ip1p=..== |

**Table 3.3:** Verb forms with lemma and inflection features ambiguities that occur in LE-PAROLE more than 20 times.

| Pers-Num | Nominative | Accusative | Dative | Reflexive | Oblique |
|---|---|---|---|---|---|
| 1st -sing | eu | *me* | *me* | *me* | mim, comigo |
| 2nd -sing | tu | *te* | *te* | *te* | ti, contigo |
| 3rd -sing | *ele*, *ela* | o, a | lhe | *se* | (*ele*, *ela*) si, consigo |
| 1st -plur | *nós* | *nos* | *nos* | *nos* | *nós*, connosco |
| 2nd -plur | *vós* | *vos* | *vos* | *vos* | *vós*, connvosco |
| 3rd -plur | *eles*, *elas* | os, as | lhes | *se* | (*eles*, *elas*) si, consigo |

**Table 3.4:** Personal pronouns system in European Portuguese.

MARv4 deals with two kinds of ambiguities. The first ambiguity is between accusative, dative and reflexive pronouns, here called ADR ambiguity, for pronouns *me, te, nos, vos*. The next sentences show examples of this ambiguity type for the 2nd person-singular form *te* 'you/yourself'.

- *No rio **te** conheci* 'In rio I met *you*'

  In this sentence *te* is the direct complement of *conheci* 'met' in the accusative case.

- *Eu dou-**te** um presente* 'I give *you* a gift'

  In this sentence *te* is the indirect complement of *dou* 'give' in the dative case;

- *Chegaste a interessar-**te*** 'Did you got *yourself* interested'

  In this sentence *te* is a reflexive pronoun.

| | me | te | nos | vos | total |
|---|---|---|---|---|---|
| **acc** | 360 | 303 | 101 | 433 | 1,197 |
| **dat** | 559 | 269 | 110 | 735 | 1,673 |
| **ref** | 316 | 186 | 54 | 28 | 584 |
| **total** | 1,235 | 758 | 265 | 1,196 | 3,454 |

**Table 3.5:** Training *corpus* for ADR class ambiguity.

The training *corpus* for these pronouns is composed by 3,454 sentences, randomly extracted from an on-line version of *Público*[1] newspaper, each containing the target pronouns. Table 3.5 maps occurrences of ADR class ambiguity on the training *corpus*.

The second ambiguity that MARv4 deals with is between nominative (subject) and oblique (prepositional) pronouns, here called NO ambiguity, for pronouns *ele, ela, nós, vós, eles, elas*, which is exemplified in the sentences below:

- *Ele fez* 'He did'

  In this sentence *ele* is the subject of *fez* 'did' and is a nominative pronoun;

- *Eu gosto de_ele* 'I am fond of him'

  In this sentence *ele* is the prepositional object of *gosto* 'foond' and is an oblique pronoun.

A *corpus* of 789 sentences is used to train the model.

Besides pronoun case disambiguation, the previous version of MARv4 was also able to choose the lemma of 36 verb forms with two admissible lemmas. In the current version of the system more 10 verb forms were added: *dito, ditas, gere, param, revisto, vi, vir, virá, vistas, viram*. The training *corpus* was developed at L2F-Spoken Language Laboratory at INESC-ID Lisboa and there is a distinct *corpus* for each verb form. Table 3.6 shows the distribution of the 47 verb forms among their possible lemmas in the training *corpus*. These verb forms were chosen because they are very frequent in CETEMPúblico *corpus* [24].

Both for lemma disambiguation and for ADR/NO ambiguity in pronouns, the models were trained with the training *corpus* processed by STRING, specifically by LexMan and RuDriCo2, which assign the admissible tags for each token.

## 3.2 Evaluation *Corpus*

From all the disambiguation mentioned above, performed by MARv4, LE-PAROLE is the evaluation *corpus*. A 10-fold cross validation strategy is used, as the training and evaluation *corpus* are the same in verbal inflection disambiguation.

[1] https://www.publico.pt/

| Verb form | Lemma 0 | Lemma 1 | # Lemma 0 | # Lemma 1 | Total |
|---|---|---|---|---|---|
| *aposta* | *apostar* | *apor* | 1,063 | 9 | 1,072 |
| *aposto* | *apostar* | *apor* | 929 | 161 | 1,090 |
| *cobre* | *cobrar* | *cobrir* | 7 | 500 | 507 |
| *cobrem* | *cobrar* | *cobrir* | 12 | 369 | 381 |
| *criam* | *crer* | *criar* | 0 | 396 | 396 |
| *descendo* | *descer* | *descender* | 408 | 2 | 410 |
| *dita* | *ditar* | *dizer* | 309 | 358 | 667 |
| *dito* | *ditar* | *dizer* | 1 | 6,814 | 6,815 |
| *ditas* | *ditar* | *dizer* | 4 | 84 | 88 |
| *entrava* | *entrar* | *entravar* | 1,113 | 30 | 1,143 |
| *foi* | *ir* | *ser* | 173 | 2,465 | 2,638 |
| *fomos* | *ir* | *ser* | 476 | 813 | 1,289 |
| *for* | *ir* | *ser* | 93 | 1,690 | 1,783 |
| *fora* | *ir* | *ser* | 14 | 700 | 714 |
| *foram* | *ir* | *ser* | 106 | 1,602 | 1,708 |
| *forem* | *ir* | *ser* | 95 | 1,278 | 1,373 |
| *fores* | *ir* | *ser* | 596 | 668 | 1,264 |
| *formos* | *ir* | *ser* | 521 | 593 | 1,114 |
| *fosse* | *ir* | *ser* | 104 | 1,519 | 1,623 |
| *fossem* | *ir* | *ser* | 56 | 961 | 1,017 |
| *fôssemos* | *ir* | *ser* | 455 | 1,094 | 1,549 |
| *fosses* | *ir* | *ser* | 257 | 693 | 950 |
| *fui* | *ir* | *ser* | 529 | 852 | 1,381 |
| *gere* | *gerar* | *gerir* | 214 | 298 | 512 |
| *lida* | *lidar* | *ler* | 650 | 609 | 1,259 |
| *morta* | *matar* | *morrer* | 192 | 437 | 629 |
| *mortas* | *matar* | *morrer* | 1,104 | 822 | 1,926 |
| *morto* | *matar* | *morrer* | 209 | 124 | 333 |
| *mortos* | *matar* | *morrer* | 214 | 99 | 313 |
| *param* | *parar* | *parir* | 67 | 0 | 67 |
| *revisto* | *rever* | *revistar* | 232 | 0 | 232 |
| *sentem* | *sentir* | *sentar* | 433 | 5 | 428 |
| *sente* | *sentir* | *sentar* | 471 | 6 | 477 |
| *sentes* | *sentir* | *sentar* | 18 | 646 | 664 |
| *tende* | *ter* | *tender* | 4 | 397 | 401 |
| *tendo* | *ter* | *tender* | 462 | 1 | 463 |
| *vendo* | *ver* | *vender* | 905 | 64 | 969 |
| *vi* | *ver* | *vir* | 2,277 | 0 | 2,277 |
| *vimos* | *ver* | *vir* | 1,505 | 234 | 1739 |
| *vira* | *ver* | *vir* | 382 | 470 | 852 |
| *virá* | *vir* | *virar* | 613 | 0 | 613 |
| *viram* | *ver* | *virar* | 766 | 43 | 809 |
| *virmos* | *ver* | *vir* | 709 | 446 | 1,155 |
| *vir* | *ver* | *vir* | 14 | 1,135 | 1,149 |
| *vista* | *ver* | *vestir* | 776 | 18 | 794 |
| *visto* | *ver* | *vestir* | 1,143 | 5 | 1,148 |
| *vistas* | *ver* | *vestir* | 972 | 1 | 973 |

**Table 3.6:** Distribution in the training *corpus*, of verb forms with lemma ambiguity, which MARv4 is able to disambiguate.

| | me | te | nos | vos | total |
|---|---|---|---|---|---|
| **acc** | 104 | 65 | 33 | 6 | 208 |
| **dat** | 227 | 105 | 57 | 11 | 400 |
| **ref** | 114 | 82 | 49 | 0 | 245 |
| **total** | 445 | 252 | 139 | 17 | 853 |

**Table 3.7:** Distribution of ADR ambiguity in the evaluation *corpus*.

| | ele | ela | nós | vós | ele | elas | total |
|---|---|---|---|---|---|---|---|
| **nom** | 274 | 191 | 90 | 0 | 54 | 16 | 625 |
| **obl** | 140 | 84 | 5 | 2 | 84 | 24 | 330 |
| **total** | 414 | 275 | 95 | 2 | 138 | 49 | 964 |

**Table 3.8:** Distribution of NO ambiguity in the evaluation *corpus*.

Tables 3.7 and 3.8 show the distribution of ADR and NO ambiguity, respectively, in the evaluation *corpus*. On the other hand, Table 3.9 shows the occurrences of ambiguous lemma verbs that MARv4 disambiguates, in the evaluation *corpus*.

| Verb form | Lemma 0 | Lemma 1 | # Lemma 0 | # Lemma 1 | Total |
|---|---|---|---|---|---|
| *aposta* | *apostar* | *apor* | 4 | 0 | 4 |
| *aposto* | *apostar* | *apor* | 2 | 0 | 2 |
| *cobre* | *cobrar* | *cobrir* | 0 | 3 | 3 |
| *cobrem* | *cobrar* | *cobrir* | 0 | 1 | 1 |
| *criam* | *crer* | *criar* | 0 | 0 | 0 |
| *descendo* | *descer* | *descender* | 1 | 0 | 1 |
| *dita* | *ditar* | *dizer* | 1 | 3 | 3 |
| *dito* | *ditar* | *dizer* | 0 | 15 | 15 |
| *ditas* | *ditar* | *dizer* | 0 | 0 | 0 |
| *entrava* | *entrar* | *entravar* | 1 | 0 | 1 |
| *foi* | *ir* | *ser* | 77 | 778 | 785 |
| *fomos* | *ir* | *ser* | 1 | 2 | 3 |
| *for* | *ir* | *ser* | 2 | 32 | 34 |
| *fora* | *ir* | *ser* | 6 | 33 | 39 |
| *foram* | *ir* | *ser* | 15 | 194 | 209 |
| *forem* | *ir* | *ser* | 0 | 11 | 11 |
| *fores* | *ir* | *ser* | 0 | 1 | 1 |
| *formos* | *ir* | *ser* | 1 | 0 | 1 |
| *fosse* | *ir* | *ser* | 6 | 78 | 84 |
| *fossem* | *ir* | *ser* | 6 | 14 | 20 |
| *fôssemos* | *ir* | *ser* | 2 | 1 | 3 |
| *fosses* | *ir* | *ser* | 0 | 4 | 4 |
| *fui* | *ir* | *ser* | 16 | 12 | 28 |
| *gere* | *gerar* | *gerir* | 0 | 1 | 1 |
| *lida* | *lidar* | *ler* | 1 | 2 | 3 |
| *morta* | *matar* | *morrer* | 1 | 0 | 1 |
| *mortas* | *matar* | *morrer* | 3 | 0 | 3 |
| *morto* | *matar* | *morrer* | 4 | 0 | 4 |
| *mortos* | *matar* | *morrer* | 3 | 0 | 3 |
| *param* | *parar* | *parir* | 0 | 0 | 0 |
| *revisto* | *rever* | *revistar* | 0 | 0 | 0 |
| *sentem* | *sentir* | *sentar* | 3 | 0 | 3 |
| *sente* | *sentir* | *sentar* | 6 | 0 | 6 |
| *sentes* | *sentir* | *sentar* | 3 | 0 | 3 |
| *tende* | *ter* | *tender* | 0 | 2 | 2 |
| *tendo* | *ter* | *tender* | 67 | 0 | 67 |
| *vendo* | *ver* | *vender* | 10 | 0 | 10 |
| *vi* | *ver* | *vir* | 22 | 0 | 22 |
| *vimos* | *ver* | *vir* | 2 | 3 | 5 |
| *vira* | *ver* | *vir* | 4 | 0 | 4 |
| *virá* | *vir* | *virar* | 1 | 0 | 1 |
| *viram* | *ver* | *virar* | 12 | 0 | 12 |
| *virmos* | *ver* | *vir* | 1 | 0 | 1 |
| *vir* | *ver* | *vir* | 3 | 45 | 48 |
| *vista* | *ver* | *vestir* | 4 | 0 | 4 |
| *visto* | *ver* | *vestir* | 32 | 0 | 32 |
| *vistas* | *ver* | *vestir* | 0 | 0 | 0 |

**Table 3.9:** Distribution in the evaluation *corpus*, of verb forms with lemma ambiguity, which MARv4 is able to disambiguate.

# 4

# MARv4 Architecture

## Contents

This chapter describes the architecture of MARv4 [21], detailing all its components, namely (i) the disambiguation of category and subcategory for a given token, (ii) the disambiguation of the lemma of particular verb forms, (iii) the disambiguation of case of some personal pronouns and, (iv) the disambiguation of inflection features in the tag of verb forms. The core of this project is the last step, however, all the chain will be described as some modifications were done in each step. Moreover, the verbal inflection disambiguation depends on the previous disambiguation processes in the chain, so it is important to understand how they work.

## 4.1 Disambiguation of category and subcategory

As previously mentioned in the Section 1.1, MARv4 is responsible for POS tagging disambiguation, selecting the best category and subcategory for the tokens. To accomplish that, it has a probabilistic HMM disambiguator, which finds the most likely sequence of tags $t_1$,...,$t_n$ to an input word sequence $w_1$,...,$w_n$. To choose the tag sequence, it uses the Viterbi algorithm [12] and its language model is based on trigrams and unigrams, according to the following equation:

$$argmax_{t_1,\ldots,t_n} \prod_{n=1}^{n} P(t_i|t_{i-2},t_{i-1})P(w_i|t_i) \tag{4.1}$$

In Equation 4.1, trigrams represent contextual information, codified in $P(t_i|t_{i-2},t_{i-1})$, where $t_i$ is the tag of the target word, $t_{i-1}$ is the tag of the previous word and $t_{i-2}$ is the tag of the word before that. Unigrams represent lexical information, codified in $P(w_i|t_i)$, where $w_i$ is the target word and $t_i$ is the tag being assigned to this word.

To support the task of creating the language model, MARv4 has a set of scripts that calculates unigrams, bigrams, trigrams, and their probabilities. As previously mentioned in Chapter 3, the manually annotated *corpus* LE-PAROLE was used to train these models.

Afterwards, the probabilistic disambiguator receives an annotated word sequence as input, with all possible tags for each word. Each tag which includes the lemma; category and subcategory; inflection features; and syntactic and semantic features. These tags are produced by the previous modules of STRING, LexMan [32] and RuDriCo2 [10]. With that information loaded, the disambiguator is then able to assign the most adequate tag to the word sequence, considering only the category and subcategory of tags, using the Viterbi algorithm.

Although this module was already working, this project required some improvements be made to it. Nevertheless, a problem was found when evaluating the system several times under the same conditions. The problem was the calculation of lexical probabilities for the words that did not occur in the

training data. Instead of using the threshold probability for nonexistent words on the training data, it was assuming those words exist and indexing the first probability (of an existent word) in a certain hashmap. However, each execution of MARv4 produced a different order in this hashmap, causing the usage a different probability for the nonexistent words in each execution and, therefore, causing the mentioned nondeterministic behavior of MARv4. This problem was solved and MARv4 became deterministic.

## 4.2 Disambiguation of verb lemmas and case of personal pronouns

As previously specified in Chapter 3, MARv4 [21] is able to disambiguate the lemma of some particular verb forms and the case for pronouns. To achieve that, MARv4 has a ML disambiguator module, which has two main phases: training and prediction.

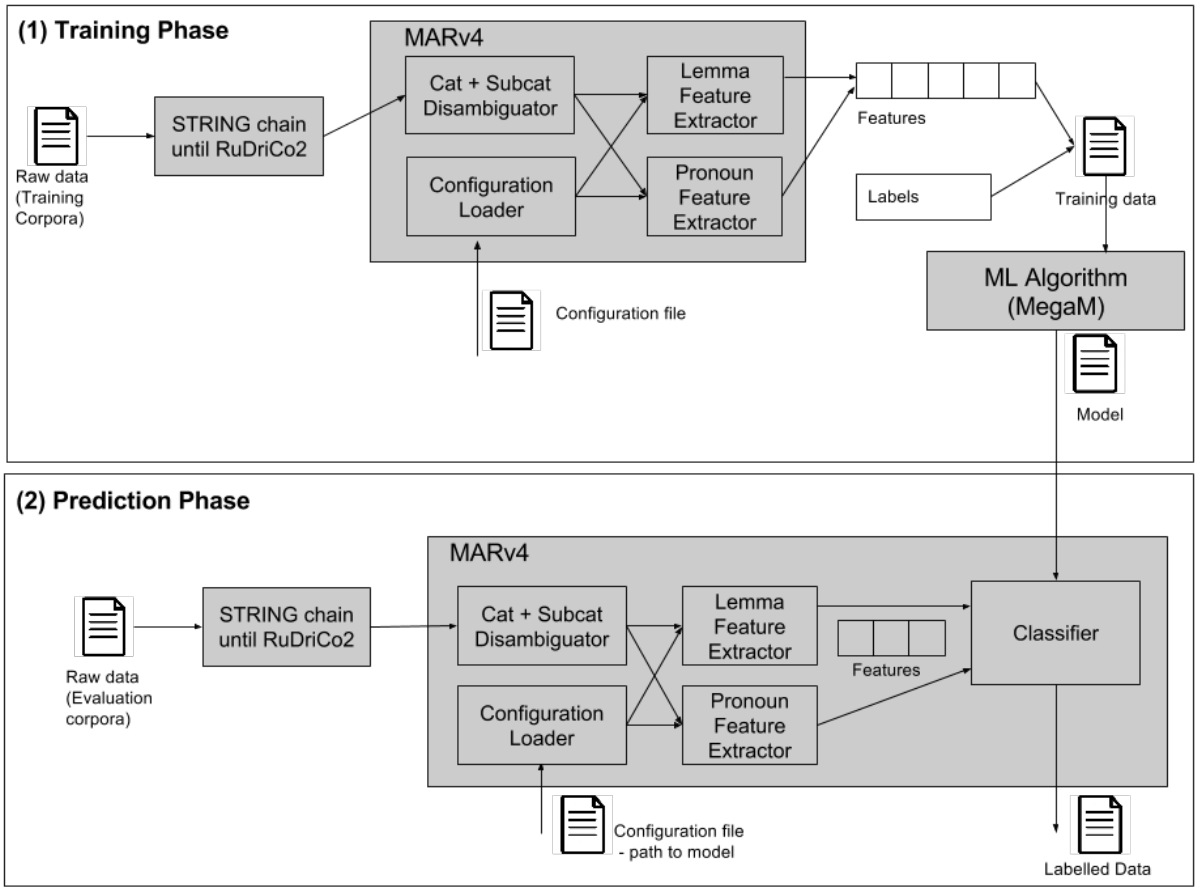


**Figure 4.1:** Training and prediction steps of MARv4, considering the lemma ambiguity of verbs and the case ambiguity of personal pronouns.

As Figure 4.1 shows, there are five major modules in the ML disambiguator: two feature extractor modules, the ML Algorithm and the Classifier. Lemma Feature Extractor, Pronoun Feature Extractor

and Classifier are implemented in MARv4. In the same figure, it is possible to observe that MARv4 is responsible for outputting either training or the labeled data, which can be chosen by an option flag.

MARv4 has a Lemma Feature Extractor and a Pronoun Feature Extractor, which extract the features regarding verbal lemma and pronoun case disambiguation, respectively. MARv4 has also another component, the Configuration Loader, which reads two types of configuration files. In the first one, the file lists all the verbs that MARv4 can disambiguate. For each verb form, the file contains the corresponding possible lemmas and the path to the model. In the second case, the two classes of ambiguity of pronouns, NO and ADR, are defined in the file. For each class, it contains the surface form of the pronouns, the possible cases, and also the path to the model for the corresponding class.

So in the training phase, STRING is executed as many times as the number of verb forms to disambiguate their lemma or the number of pronoun ambiguity classes. So, in this phase, STRING processes the training *corpus*. Therefore, MARv4 receives the training corpora annotated with the possible tags for each token in the segments. After that, the Cat + Subcat Disambiguator selects the best tag for each token, considering only the category and subcategory, using Viterbi algorithm. Afterward, one of the feature extractor modules computes all the features, which are detailed further ahead in this section. Finally, MARv4 outputs the features, which are then concatenated with the respective labels. An instance of the result training data is obtained in Expression 4.2. In the expression *class* is a numeric value which codifies the verbal lemma (in lemma ambiguity) or the case of the target pronoun (in ADR/NO ambiguity). *FEATURE*$_n$ is the name of feature $n$ in the instance, and it is followed by its value and weight.

$$class \quad FEATURE_1 : value \quad weight \quad FEATURE_2 : value \quad weight \quad FEATURE_n : value \quad weight \tag{4.2}$$

With respect to verb forms with ambiguous lemmas, the Feature Extractor (see Figure 4.1) picks a window size of 5. Therefore, considering that the verb form to disambiguate is the word $w_i$, the sequence of neighbour words $w_{i-2}$, $w_{i-1}$, $w_{i+1}$ and $w_{i+2}$ are analysed, producing the following features for each word:

- `WORD`: surface of the neighbour word;
- `POS`: category and subcategory of the neighbour word or, if its category is Verb, the value for this feature is the mood of the verb form.

For pronouns, the features and the window size depend on the ambiguity class. For ADR ambiguity class, Feature Extractor also picks a window of 5 words, where the pronoun to disambiguate is the word $w_i$ and the neighbouring words are $w_{i-2}$, $w_{i-1}$, $w_{i+1}$ and $w_{i+2}$; the features extracted are:

- `WORD`: the form of pronoun $w_i$;

- `LEMMA:` the lemma of the verb;
- `CLITIC?:` the *clitic* use of the pronoun, that is, whether it is attached to a verb or not (values: yes/no);
- `NOTREFLEX?:` a feature based on the person-number inflectional values of both the pronoun and the verb, indicating whether they are equal or not (values: yes/no);
- `VERBPOS:` mood-tense inflectional values of the verbs;
- `PNG:` the person-number and gender inflectional features of the verb;
- `VIPER:` the syntactic and semantic class of the verb[1], where each class represents a verb construction type, which can be seen as one of the verb-senses a verb may have (values: 71 verb syntactic-semantic classes).

For NO ambiguity class, Feature Extractor has a window size of 3, where the pronoun to disambiguate is the word $w_i$ and the neighboring words are $w_{i-1}$ and $w_{i+1}$; the features extracted are

- `WORD:` the form of pronoun $w_i$;
- `LEMMAPROX:` the lemma of the neighboring words;
- `CAT + SUBCAT:` the POS, both the main category and the subcategory of the neighboring words;
- `VERBPOS:` mood-tense inflectional values of the verbs;
- `PNG:` the person-number and gender inflectional features of the verb;

At last, as shown in Figure 4.1, MARv4 uses ML algorithm, provided by an an external tool MegaM[2], to train a ME-based model. This tool is an implementation of conjugate gradient, for binary problems as disambiguation of verbal lemmas, and limited memory BFGS for multiclass problems [8], like the disambiguation of case for personal pronouns.

Some sentences of the training *corpus* may be not be used to train the models. This is due to the fact that the sentences are processed by STRING and, consequently some of the verbs or pronouns may have been incorrectly classified as to their category.

In the second step, the prediction phase, the evaluation *corpus* is processed by the modules of STRING that precede MARv4, and all the tags are assigned to each token as in the training phase, so that Cat + Subcat Disambiguator (see Figure 4.1) may assign the best tag, considering only the category and subcategory. Then, when a target pronoun or verb form is processed, the corresponding features are extracted and the classifier uses them together with the corresponding model, loaded by

[1] Described in Baptista(2012) [2].
[2] http://www.umiacs.umd.edu/ hal/megam/

Classification Loader, to assign the best class which will be the lemma of the verb or the case of the pronoun.

The main contribution to improve this module consisted in the revision and correction of the Feature Extractor modules. Moreover, 10 new verb forms were added to the lemma disambiguation, as detailed in Chapter 3. On the other hand, some details regarding the training *corpus* were analysed. The module that chooses the category of the tokens can assign, for instance, the noun category to a verb form, which MARv4 is considering for lemma disambiguation. In this case, features will not be extracted and the sentence is not included in the model. The same applies if a pronoun is classified with another category.

## 4.3 Disambiguation of verbal inflection features

In the current version of the system, STRING is already able to disambiguate some particular cases of verbal inflection ambiguity, through its rule-driven morpho-syntactic disambiguator, named RuDriCo2 [10]. However, one of the objectives of this work is to implement a full disambiguation of verbal inflection features, adopting a statistical approach based on ML algorithms. This subsection describes the most relevant implementation details of the inflection features disambiguator for verb forms developed in this project.

### 4.3.1 Building the annotated training data

There is no specific annotated training *corpus* for verbal inflection features. Therefore, LE-PAROLE was used, due to the effort required for annotating a new corpus.

To build the training data two modules were developed. The first module is the Corpus XML Reader (see Figure 4.2), which will read a manually annotated *corpus* in XML format. The second module is the Context Generator, which has a Feature Extractor and a Configuration Loader, and it is used to generate all the features for each verb form in the *corpus* and to read some relevant configurations in order to generate the training data, as in the verbal lemma and pronoun case disambiguation. Following Expression 4.2, each instance of the training data refers to each verb form in the training *corpus* and it is composed of the features extracted (detailed in Section 4.3.3), followed by their weight (which always assume the value 1). The first element of an instance is the class of the verb form. Computing the class is not a straightforward process, therefore, it is explained in more detail in the Section 4.3.2.

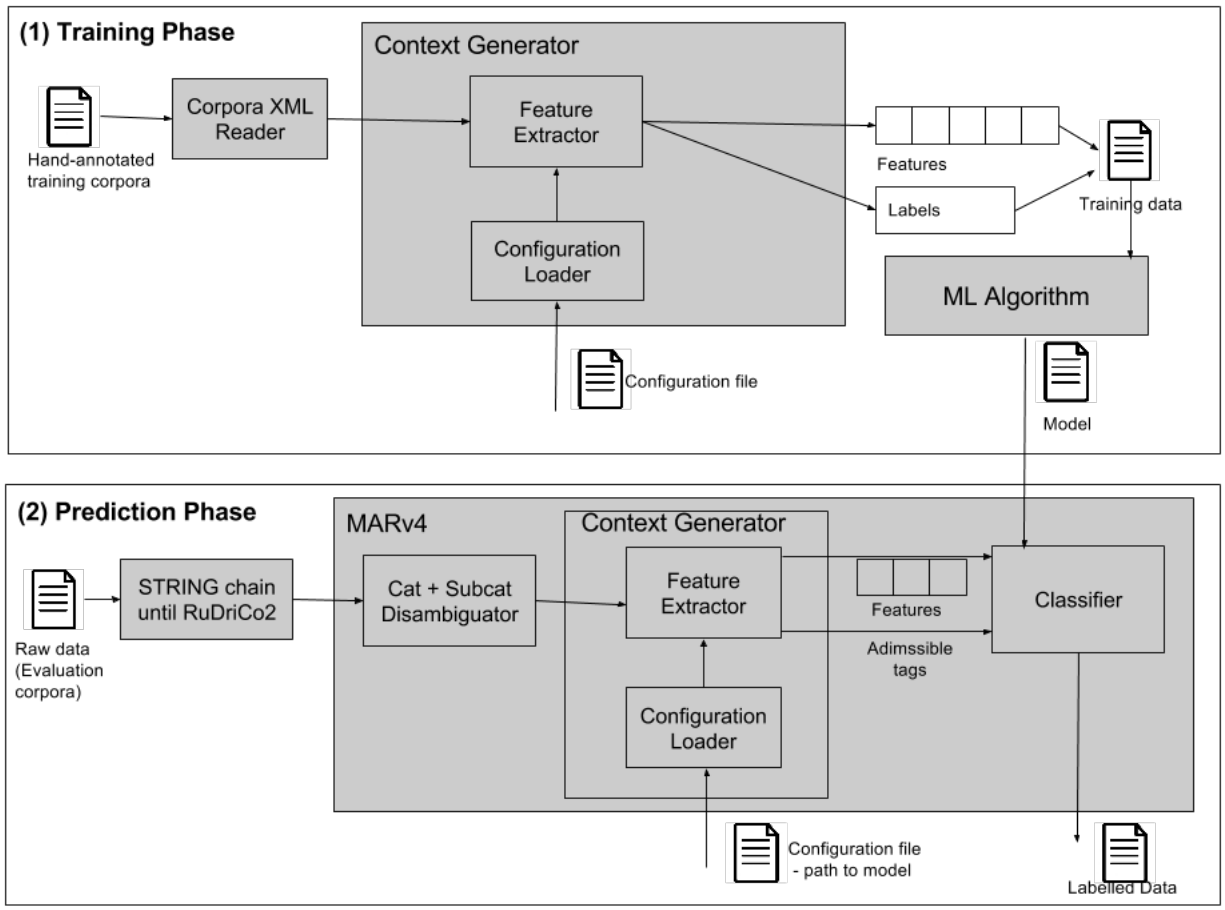


**Figure 4.2:** Training and prediction steps of the system, considering the verbal inflection features disambiguation.

### 4.3.2 Computing classes

Since a verb form has several inflection features: mood (modality), tense, person, number and gender; disambiguating its tag in view of its inflectional information can be achieved in one step or it can be achieved using a sequential process. In a sequential process, each element in the sequence is a verbal inflection feature or a combination of them. Therefore, each component in the sequence corresponds to a file containing its own training data, which is the input file of the ML Algorithm (Figure 4.2) that generates the models. So, the class is codified using one of the following options:

- **Single step process:** the class codifies information about all the values of all inflection features presented in the verb tag;
- **Sequential process:** the class codifies the value of a single inflection feature or a combination of inflection features presented in the verb tag.

A configuration file defines which method to use and it is loaded by the Configuration Loader (see Figure 4.2). Each line of this file corresponds to one training data file, generated outside of MARv4 (see

Figure 4.3). Each line contains: the inflection feature(s), followed by its (their) position(s) in the tag, according to the STRING tagset; the index of the features that will be used; finally the path where the trained model will be saved. The order of the lines correspond to the order the inflection feature(s) is (are) disambiguated, as it will be further explained in Section 5.5. This example is a representation of the sequential method as there is a trained model per inflection feature. The order of the disambiguation is first the mood, followed by tense, person and number inflection features. Next, in the same file, there is a correspondence between the index of all features and its name.

```
#Verb feature : features position : model features : path to model

m : 3 : 1 2 3 4 5 : ./models/m.txt
t : 4 : 1 2 3 4 5 : ./models/t.txt
p : 5 : 1 2 3 4 5 : ./models/p.txt
n : 6 : 1 2 3 4 5 : ./models/n.txt

#index : feature name
1 : POS-1-CATSUBCAT
2 : POS1-CATSUBCAT
3 : NOUNBEFORE
4 : NOUNBEFORENUM
5 : ADJBEFORE
```

**Figure 4.3:** Configuration file used in the disambiguation of verbal inflection features, through a sequential disambiguation process, supposing there are only 5 features.

The feature extraction process is crucial in both training and prediction phases, and it will be specified in the next section.

### 4.3.3 Extracting features

In the process of verbal inflection features disambiguation, it is possible to use different features for the classifier. Thus, these features are defined in the configuration file (see Figure 4.3 in Section 4.3.2), and the Configuration Loader (see Figure 4.2) is responsible to map this information to proper feature names defined in the same file.

The set of features, which are generated by the Feature Extractor (see Figure 4.2), for each verb form in corpora, as the result of a selective process. Table 4.1 describes the set of features used to develop the proposed system. A feature codifies information about a neighbouring word of the target verb $w_i$. The neighbour words are defined in the Window column, followed by the name of the feature and its description. The Relation column indicates that the features with the same number have a relation, which is considered in the feature selection process, described in Section 5.5.6. This relation indicates that the set of features consider the same neighbour word.

If none of the neighbouring words of the target verb form (defined in the Window column) meet the requirements in the feature description, it will originate a missing value in the instance for that feature and nothing will be output to training data. The same order of features is always guaranteed.

As Table 4.1 describes most of the features are related to personal pronouns in the neighbourhood of the target verb. Therefore, it is important to note that the disambiguation of the case of these pronouns has an impact on verbal inflection disambiguation, as their case values are features of these models (PPCASEBEFORE and PPCASEAFTER in Table 4.1).

| Window | Feature Name | Feature Description | Relation |
|---|---|---|---|
| $w_{i-1}$ | POS-1-CATSUBCAT | Category and subcategory of word $w_{i-1}$. | 1 |
| $w_{i+1}$ | POS1-CATSUBCAT | Category and subcategory of word $w_{i+1}$. | 2 |
| $\{w_{i-1}, w_{i-2}, w_{i-3}\}$ | NOUNBEFORE | Distance of the first noun to the target verb (value 1, 2 or 3). | 3 |
| | NOUNBEFORENUM | Number value of the first noun to the target verb, among the words in the window. | |
| | ADJBEFORE | Distance of the first adjective to the target verb (value 1, 2 or 3). | 4 |
| | ADJBEFORENUM | Number value of the first adjective, among the words in the window. | |
| | PPBEFORE | Distance of the first personal pronoun to the target verb form (value 1, 2 or 3). | 5 |
| | PPCASEBEFORE | Case value of the first personal pronoun, among the words in the window. | |
| | PPREFLEX | Distance of the first personal pronoun with reflex case to the target verb form (value 1, 2 or 3). | 6 |
| | PPREFLEXPER | Person value of the first personal pronoun with reflex case, among the words in the Window. | |
| | PPREFLEXNUM | Number value of the first personal pronoun with reflex case, among the words in the Window. | |
| | PPNOTREFBEFORE | Distance of the first personal pronoun, which is not reflex, to the target verb form (value 1, 2 or 3). | 7 |
| | PPNOTREFBEFOREPER | Person value of the first personal pronoun which is not reflex, among the words in the Window. | |
| | PPNOTREFBEFORENUM | Number value of the first personal pronoun which is not reflex, among the words in the Window. | |
| | PPNOBEFORE | Distance of the first personal pronoun with nominative or oblique case, to the target verb form (value 1, 2 or 3). | 8 |

| Window | Feature Name | Feature Description | Relation |
|---|---|---|---|
| $\{w_{i-1}, w_{i-2}, w_{i-3}\}$ | PPNOBEFOREPER | Person value of the first personal pronoun with nominative or oblique case, among the words in the window. | |
| | PPNOBEFORENUM | Number value of the first personal pronoun with nominative or oblique case, among the words in the window. | |
| $\{w_{i+1}, w_{+2}, w_{i+3}\}$ | PPAFTER | Distance of the first personal pronoun to the target verb form (value 1, 2 or 3). | 9 |
| | PPCASEAFTER | Case value of the first personal pronoun, among the words in the window. | |
| | PPCLITIC | Distance of the first personal pronoun with clitic to the target verb form (value 1, 2 or 3). | 10 |
| | PPCLITICPER | Person value of the first personal pronoun with clitic, among the words in the window. | |
| | PPCLITICNUM | Number value of the first personal pronoun with clitic, among the words in the window. | |
| | PPNOTREFATER | Distance of the first personal pronoun not reflex to the target verb form (1, 2 or 3). | 11 |
| | PPNOTREFAFTERPER | Person value of the first personal pronoun not reflex, among the words in the window. | |
| | PPNOTREFAFTERNUM | Number value of the first personal pronoun not reflex, among the words in the window. | |
| | PPNOAFTER | Distance of the first personal pronoun with nominative or oblique case to the target verb form (value 1, 2 or 3). | 12 |
| | PPNOAFTERPER | Person value of the first personal pronoun, with nominative or oblique case, among the words in the window. | |
| | PPNOAFTERNUM | Number value of the first personal pronoun, with nominative or oblique case, among the words in the window. | |
| $\{w_0, ..., w_{i-1}\}$ | SUBCONJ | Position of the first subordinating conjunction which appears before the target verb without any other verbs between the two words. | 13 |
| | LEMMASUBCONJ | Lemma of the first subordinating conjunction, among the words in the window, without any other verbs between the subordinating conjunction and the target verb form. | |

| Window | Feature Name | Feature Description | Relation |
|---|---|---|---|
| $\{w_0, ..., w_{i-1}\}$ | VERBFINITEPER | Person value of the first finite verb, among the words in the window, without any other verbs between the finite verb and the subordinating conjunction and between the conjunction and the target verb. | 14 |
| | VERBFINITENUM | Number value of the finite verb, among the words in the window, without any other verbs between the finite verb and the subordinating conjunction and between the conjunction and the target verb. | |

**Table 4.1:** Features extracted in verbal inflection features disambiguation. The column window represent the neighbour words considered in each feature. The features with the same Relation number are dependent.

### 4.3.4 ME Classifier

In the training phase, to compute ME models for verbal inflection disambiguation, the MegaM tool is used . MARv4 has already a binary and a multiclass ME classifier, to disambiguate verbal lemmas and the case of personal pronouns. In the developed approaches, the classifier has to decide the most adequate class for each instance, among all the classes in the respective model. However, for verbal inflection disambiguation, this process is slightly different and a new ME classifier had to be developed.

A verb form has a variable number of admissible tags, assigned by previous modules of STRING. Therefore, the classifier has to decide the class, only among this subset of tags, instead of considering all the classes presented in the trained model. Therefore, the ME classifier receives the features and also the admissible tags available for the target verb form. Then, it computes the maximum likelihood only for the classes that codify the verbal inflection features previously defined for the current word disambiguation.

### 4.3.5 Naive Bayes Classifier

This section describes the implementation of Naive Bayes and its integration in MARv4. Naive Bayes algorithm [23] is a supervised learning algorithm that is based on Bayes theorem and that assumes the independence between all the features. The classifier decides the class of an instance according to Equation 4.3.

$$classify(f_1, ..., f_n) = \underset{c}{\operatorname{argmax}}\, p(C = c) \prod_{i=1}^{n} p\left(F_i = f_i \,|\, C = c\right) \tag{4.3}$$

where $C$ is the class that will be determined and $F_i$ is each feature of the instance. According to the Maximum Likelihood Estimation, these individual conditional probabilities, for class a $C_j$, $p(F_i|C_j)$, is the

ratio of the occurrences of the feature $F_i$ and label $C_j$ together to the total number of occurrences of class $C_j$ in the training data. However, if a feature $F_i$ does not appear together with some class $C_j$ in the training data, the add-one smoothing will be applied to obtain the conditional probability, according to Equation 4.4.

$$psmoothed\,(F_i \,|\, C_j) = \frac{|F_i, C_j| + 1}{|C_j| + |F|} \tag{4.4}$$

Without smoothing, the probability for the features that do not appear in the training data together with a class, would be 0, which could leaved to a greater number of incorrectly classified instances.

The training data used in Naive Bayes algorithm is the same as in ME. However, to obtain the model used in the classifier, a new module were developed. This module loads all the instances of the training data and groups all the pairs feature name, value in a hash. Each {feature name, value} has a vector where the value of position $i$ corresponds to the number of occurrences of the {feature name, value} pair on the training data, in the instances annotated with class $i$. This module will then output a model where each line corresponds to a {feature name, value} pair followed the values in its vector, which contains the occurrences per class. The occurrences are separated by spaces.

Therefore, in the prediction phase, MARv4 will use the Configuration Loader module (see Figure 4.2) to load the model. As in ME, this classifier will only compute the probabilities for the classes which constitute the subset of admissible tags for the target verb.

Is is important to note that, in spite of the features used on morphosyntactic disambiguation for verb forms not being independent, Naive Bayes was implemented. This is due to the fact that this ML algorithm is fast and easy to use in order to predict the class or even for training the model. Another reason is that is it almost impossible to have a set of features which are completely independent in real life and, there are several researches works (for instance [34]) that prove that Naive Bayes can also achieve good results, even with dependent features.

### 4.3.6 Implementing MFF method

The approach proposed by Branco & Nunes [6] suggested an MFF algorithm for verbal inflection disambiguation for European Portuguese, and it obtained 95.28% of precision. Therefore, to obtain preliminary results for verbal inflection disambiguation in LE-PAROLE and, to compare these results with ML disambiguation, the MFF algorithm was implemented in MARv4. As it was explained in Chapter 2.1, this approach chooses the most frequent pair {verb form, tag}, out of the admissible tags that the verb form has been assigned in training *corpus*. If none verb does not exist in this *corpus*, this approach select the most frequent tag in the training corpora, out of the admissible tags assigned to that form by

RuDriCo2. If any of the admissible tags of the verb form occurs in the training corpora, the algorithm selects a random tag.

To compute frequencies of each pair {verb form, tag} in the training data, some scripts were developed. These scripts used the lexical information already computed for category and subcategory disambiguation, described in Section 4.1. Therefore, the training *corpus* that was used to this task is also the manually annotated *corpus* LE-PAROLE. MARv4 saves this information in two distinct hash tables: one of them is indexed by the verb form and has a corresponding map, which indexes the tags for the verb and the number of occurrences in the training data; the other hashtable is indexed by the tag and has the overall occurrences of the tag in the training corpora.

## 4.4 Conclusion

As it was described along this Chapter, MARv4 is a statistical disambiguator that performs four types of disambiguation, as shown in Figure 4.4. The sequence of these disambiguation tasks presented in the figure, corresponds to the order that achieved better results as it will be described in Chapter 5. MARv4 receives a *corpus* to evaluate, processed by the previous modules of STRING and its first task is to assign the best category and subcategory for each token in a sentence. Then, the Lemma Disambiguator processes the same sentence and if there is any Verb in the sentence, out of the lemmas, for which MARv4 has trained their respective models, the system choses the most suitable lemma for that verb. In the third step, if the sentence contains any ambiguous pronoun, the classifier assigns the case to the target pronoun. Last, for each verb in the sentence, the Verbal Inflection Features Disambiguator chooses the most convenient verbal inflection features, presented in the tag, and considering all the disambiguations made before.

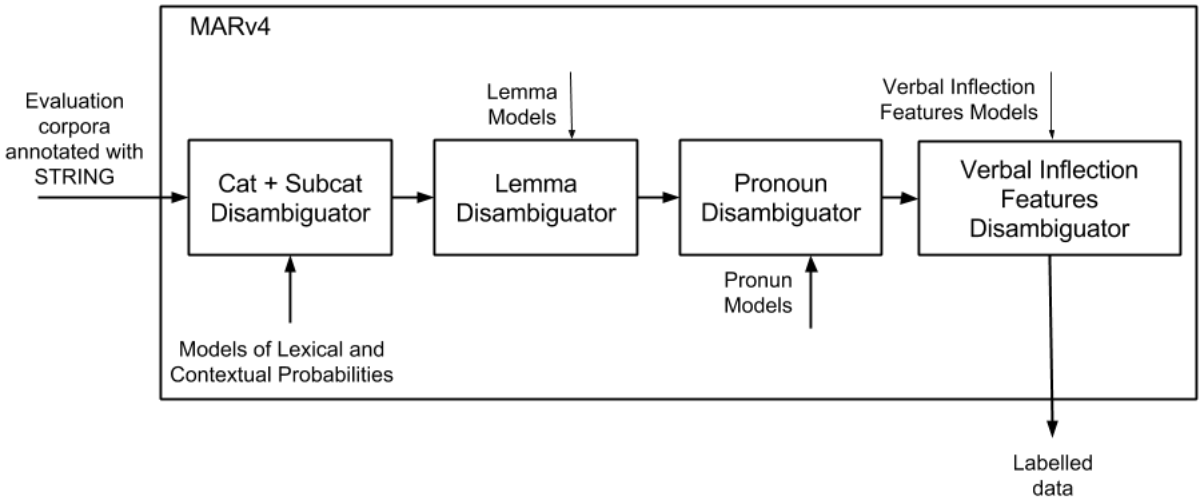


**Figure 4.4:** Sequence of disambiguation tasks performed by MARv4 in its prediction phase.

# 5

# Evaluation

## Contents

This chapter details the results for each disambiguation performed by MARv4, described in Chapter 4. As it was mentioned in Chapter 3 the LE-PAROLE *corpus* was used to evaluate all the experiments described in this chapter. The system used a 10-fold cross validation method, to randomly partition the data in ten parts of equal size, successively learning with 9/10 of the annotated material and evaluating with the models remaining fold. Results are the average of the 10 runs.

## 5.1 Measures

In order to perceive which are the best techniques to perform the different levels of disambiguation performed by MARv4, precision was used, a common measure among the state-of-the-art systems. Expression 5.1 is a formal definition of precision, where $TP$ represents the true positive values and $FP$ represents the false positive values in the evaluation *corpus*.

$$precision = \frac{TP}{TP + FP} \tag{5.1}$$

## 5.2 Measuring the effects of rules

Some tests were performed in order to observe the effects of using disambiguation rules combined with the statistical disambiguators. The module responsible for the rules mentioned above is RuDriCo2 [10], which is the module of STRING that preceeds MARv4. RuDriCo2 [10] has two types of rules. The first type changes the segmentation , either contracting several words into a token or expanding single word expressions into several expressions. The second type of rules disambiguates tokens with more than one possible tag. The code snippets below shows the two types of rules. Program 1 shows a segmentation rule. The purpose of the rule is to expand the word *aonde* 'to_where' into the words *a* 'to' and *onde* 'where', assigning the Preposition category to the first *a* and selecting two different tags to the second word *onde*, which cause a pronoun either from the interrogative or the relative subcategories.

**Program 1** Example of segmentation rule of RuDriCo2.

```
0> [surface='aonde']
  :<
   [surface='a',lemma='a',CAT='pre'],
   [surface='onde',lemma='onde',CAT='pro',SCT='itr']
                  [lemma='onde',CAT='pro',SCT='rel'].
```

Program 2 presents a disambiguation rule of RuDriCo2, which prefers Infinitive than other possible values for the mood (modality) inflection feature, when the verb appearing after comma $<,>$.

**Program 2** Example of disambiguation rule of RuDriCo2.

```
0> |[lemma=',']|
   [MOD='inf'][MOD=~'inf']
  :=
   [MOD='inf']+.
```

In this settings, three scenarios were defined to assess the impact of combining the rule-based approach with statistical disambiguation modules of MARv4.

- **No disambiguation rules:** this set contains only the rules which are able to segment the input sentences;
- **No verb disambiguation rules:** this set is composed of the previous set of rules and other rules, which disambiguate tokens, except the words that are classified as a verb. Therefore, there is no rule able to disambiguate verbal inflection features;
- **All rules:** it aggregates all the rules, including the first set of rules and the ones which disambiguate tokens, independently if the word is classified as a verb or any other category.

It is important to note that, for each set of rules, the statistical methods produce different results, even in the category disambiguation. This is due to the fact that, for instance, the second set of rules excludes some rules which operate in the verbal inflection features disambiguation. Therefore, as there are some segmentation rules that depend on verbal inflection features, the segmentation will be different, compared with the third set.

For example, Program 3 contains two segmentation rules that depends on the verbal inflection features. Thes rules are able to classify the Portuguese word *deste* 'you_gave' as a verb. In a sentence like *já te deste conta ...* 'have you realized ...' the rules read as follows: the tensed form *deste* of verb *dar* 'give' is selected when it appears immediately after an *accusative* or *dative* pronoun.

**Program 3** Example of segmentation rule which depends on verbal inflection features.

```
0> |[CAS='dat']|
   [lemma='dar',CAT='ver',MOD='ind',TEN='ppe',PER='2'][CAT='pre']
  :=
   [CAT='ver']+.

0> |[CAS='acc']|
   [lemma='dar',CAT='ver',MOD='ind',TEN='ppe',PER='2'][CAT='pre']
  :=
   [CAT='ver']+.
```

## 5.3 Disambiguation of category and subcategory

As mentioned in Section 4.1, this work produces some modifications in the category and subcategory disambiguation. Besides, Section 4.3.3 specifies that verbal inflection disambiguation depends on category and subcategory of the neighbour words of the target verb. For these reasons, this section presents the results of the disambiguation of category and subcategory and it measures the effects of the rule-based module (presented in Section 5.2) on this disambiguation. RuDriCo2 is executed with each set of rules, processing the input sequence of tokens. Then, MARv4 receives the output of RuDriCo2 and proceeds with the disambiguation of category and subcategory of the input sequence.

As Figure 5.1 exhibits, using all the disambiguation rules is the best approach for category and subcategory disambiguation. Category disambiguation achieved a precision of 98.12%, which represent a gain of 0.1% above the disambiguation with no verb disambiguation rules and 0.58% with no disambiguation rules. Considering the subcategory, the system achieved 97.75% of precision with all the rules.

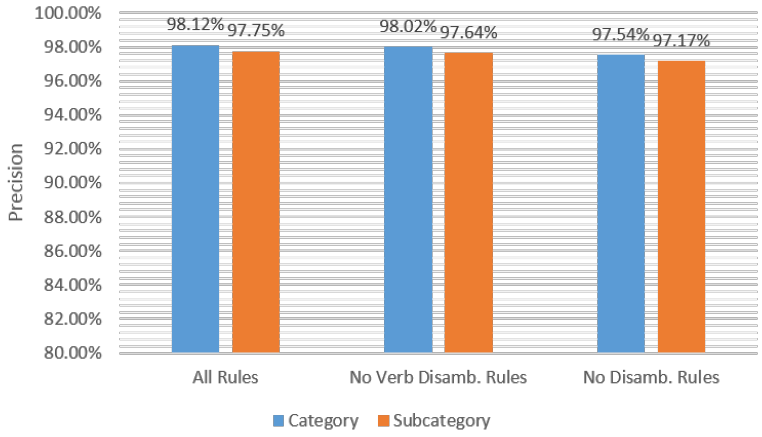


**Figure 5.1:** Category and subcategory disambiguation performed by MARv4 among the three sets of rules.

Figure 5.2 shows the precision of all the categories. Punctuation, the nouns, and verbs are the categories with better results. A precision of 99.43% is obtained for verbs and 98.76% for the nouns when the input sequence generated by RuDriCo2 with all rules is combined with the disambiguator.

Notice that the only POS where rules seem to have some impact are Adjective, Pronoun and Preposition, with a slight difference in Conjunction, where the no-rules scenario decreases 0.56%. The difference between the all-rules and no-verb rules scenarios is also almost negligible, except for Pronoun and Preposition, with all-rules scenario being a trifle better for Adjective. This means that verb disambiguation rules have a small impact against all-rules scenario.

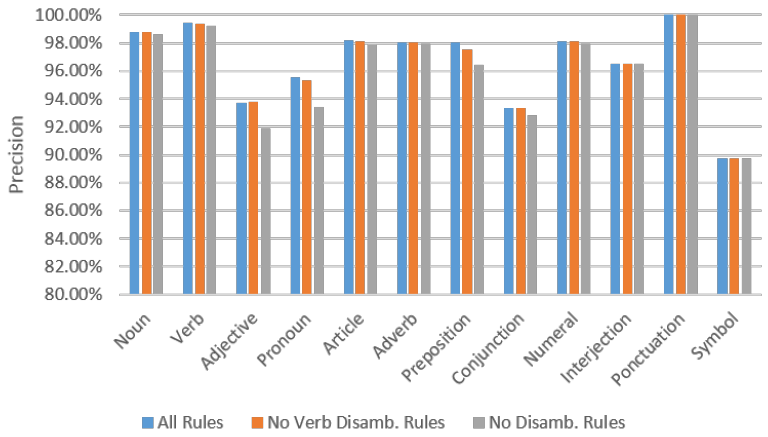


**Figure 5.2:** All the categories evaluated with the three sets of rules combined with the category and subcategory disambiguator.

## 5.4 Disambiguation of pronouns

This work adapted the process of pronoun disambiguation and as discussed in Section 4.3.3 the verbal inflection disambiguation has many features related to pronouns in the neighbourhood of the target verb, so the performance achieved for pronouns conditions the performance of verbal tags.

The experiments in this section assume the sequential disambiguation starting by category and subcategory disambiguation, followed by verbal lemma disambiguation and finally, the pronoun disambiguation is executed.

In the task of disambiguating pronouns, combining the input of RuDriCo2 with all rules, feature selection tests were performed. Figure 5.3 shows the results for the ADR ambiguity class, removing each feature from all-features set. The most relevant features are `NOTREFLEX`, `PNG` and `VIPER`, as without each one of them the system only reached a precision below 61%. The highest precision was achieved when including all features, presenting a value of 79.60%.

For the NO class ambiguity, Figure 5.4, the WORD feature is the most significant one, as without it the system obtains the lowest precision, 96.36% with a drop of almost 3% from the all-features scenario, which reached 99.17%.

The impact of the rules in the disambiguation of both ambiguity types was tested, combined with the disambiguators with all-features, as it is the approach that achieves the highest precision in both of ambiguity classes. In Figures 5.5 and 5.6 it is observed that all the disambiguation rules combined with the developed disambiguator is the best approach, regarding the disambiguation of the case for both ambiguity classes. It can also be seen the influence of the missing disambiguation rules regarding verbal inflection features, in the precision achieved by the disambiguator. Therefore, when the set of rules with no verb disambiguation rules is combined with the disambiguator, the system achieved a

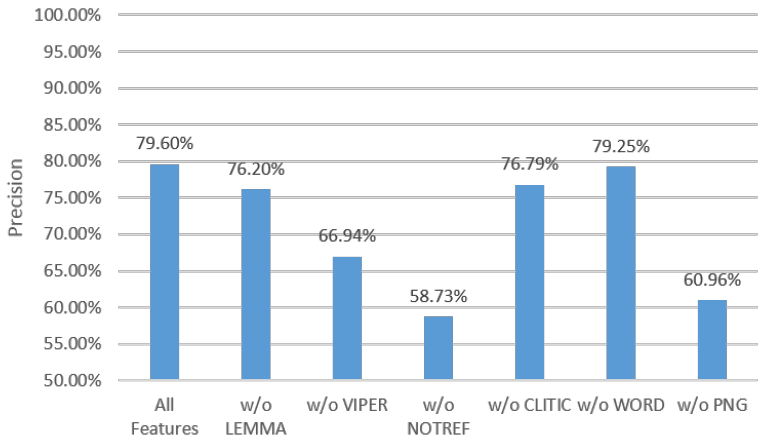


**Figure 5.3:** ADR class ambiguity for pronouns, combining the output of RuDriCo2 will all the disambiguation rules with MARv4, testing ML pronoun disambiguator without each one of the developed features.

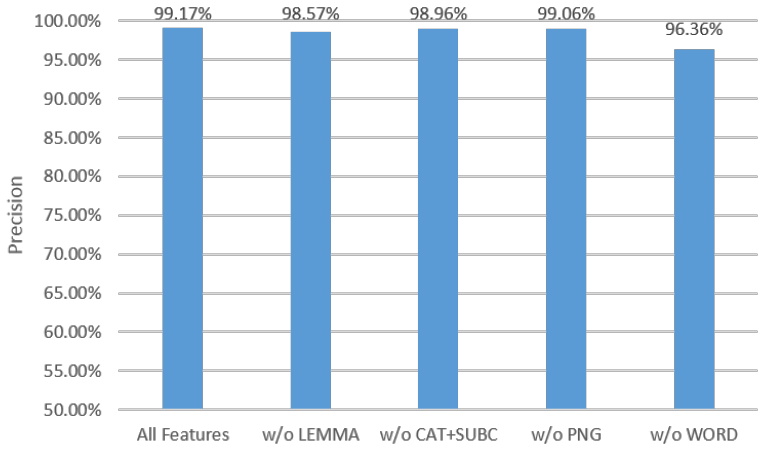


**Figure 5.4:** NO class ambiguity for pronouns, combining the output of RuDriCo2 will all the disambiguation rules with MARv4, testing ML pronoun disambiguator without each one of the developed features.

precision of 73.04% for ADR ambiguity class and 98.85% for NO ambiguity class, 6.56% and 0.32% below the best-achieved precision, respectively. One possible justification is that some features (as `PNG`) used in pronoun disambiguation depends on verbal inflection features, which can have some wrong values as no verbal disambiguation rules are used.

Additionally, it can be observed in the above-mentioned figures that the disambiguator performs better in NO ambiguity class than in ADR ambiguity class and this is possibly due to the fact that NO disambiguation is a binary classification problem against the ADR disambiguation, which is a multi-class problem, with 3 classes.

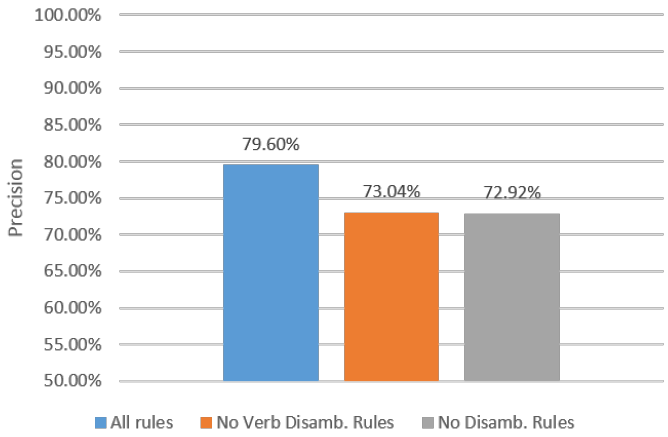


**Figure 5.5:** Results of ADR class ambiguity for pronouns, with all the set of rules combined with the disambiguator.

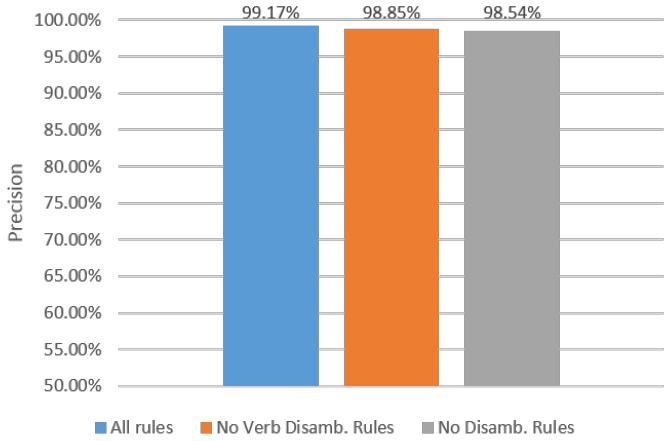


**Figure 5.6:** Results of NO class ambiguity for pronouns, with all the set of rules combined with the disambiguator.

## 5.5 Verbal inflection disambiguation

This section presents the results of the different methods developed for the verbal inflection disambiguation task. Firstly, it presents the results for the baseline, namely the previous version of MARv4. This baseline consists of the output of RuDriCo2 using all rules, combined with a sequence of disambiguation tasks, as it will be described in Section 5.5.1. The results achieved by the MFF method (described in Section 4.3.6) will also be presented.

Then, this section describes all the parametrisations tested for the several ML disambiguators, as well as the corresponding results. Finally, this section aims to compare and discussing the results achieved by all the disambiguators.

An important assumption is that the results for precision consider only the verbs which the system correctly classified as a verb, i.e. if a verb was classified with another category it will not return a wrong result for precision.

For the different disambiguators the precision is measured by way of two indicators:

- **IFtag:** the precision of selecting only verbal inflection features in a verb tag;
- **IFtag+lemma:** the precision of selecting verbal inflection features in a verb tag along with its lemma.

### 5.5.1 Baseline

Figure 5.7 shows preliminary results for verbal inflection disambiguation. The baseline consists on the output of RuDriCo2 with all rules, combined with MARv4 executing the sequence of ML techniques to perform the disambiguations, starting by category and subcategory, followed by verbal lemma disambiguation and finally, pronoun disambiguation. The set of rules does not disambiguate all the verbal tags. When none of the rules is triggered, the baseline choose the first tag from the possible tags of a given verb.

As Figure 5.7 illustrates, the baseline obtained a precision of 91.67% for the indicator IFtag and 91.25% for IFtag+lemma.

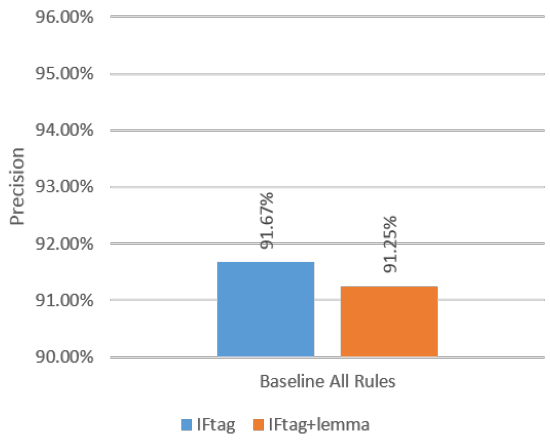


**Figure 5.7:** Results of the defined baseline for only IFtag and IFtag+lemma for verbs.

### 5.5.2 MFF experiments

This section shows the results of the output of RuDriCo2 with no verbal disambiguation rules, combined with MARv4, starting by category and subcategory disambiguation, followed by lemma disambiguation and pronoun disambiguation. Finally, MFF implemented according to Section 4.3.6, is ex-

ecuted, in order to disambiguate ambiguous verb forms regarding inflection features. As Branco & Nunes [6] proposed, MFF was also tested with a certain threshold, which means that if the most frequent sequence verb form-tag is equal or below of the threshold, it is ignored.

Figure 5.8 shows that the best result for MFF is when the verb forms-tag with three or fewer occurrences in the training data are ignored, obtained a precision of 94.78% for IFtag and 94.31% for IFtag+lemma. This represents a slightly gain when no verb form-tag are discarded (threshold 0). Notice that the difference between the threshold values shows a growing trend from 0 to 3 and becomes asymptotic at threshold 4, although the absolute difference is almost negligible (0.13 for IFtag and 0.12 for IFtag+lemma) between 1 and 4.

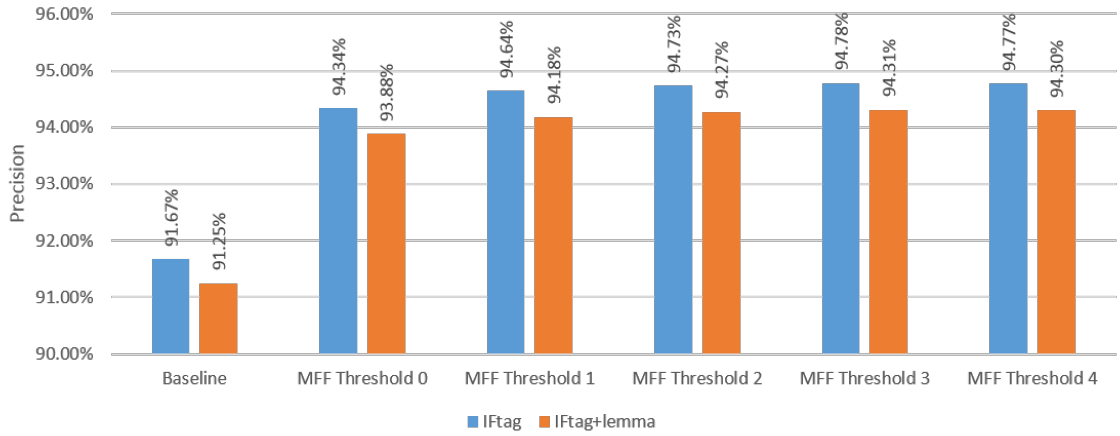


**Figure 5.8:** Results of no disambiguation verbal rules combined with MFF and the baseline with all rules, for only IFtag and IFtag+lemma.

The best result of MFF obtained an improvement over the baseline of 3.11% in the first indicator and 3.06% in the second indicator. A possible reason for this gain is that the STRING rules that disambiguate verb inflection features cover only a limited number of ambiguity cases. However, if a different corpus were to be used, the difference between the MFF results and the baseline would likely be less expressive; eventually even outperformed by the baseline, and this is due to the fact that the rules in STRING are generic in nature, so they can be more promptly ported to other *corpora*, while the MFF algorithm is always adjusted to the corpus; an indication of this can be inferred from the fact that the best approach reported by Branco & Nunes [6] for MFF on the CINTIL *corpus*, with threshold 1 and with automatic POS tag, was 96.51% (see Table 2.3). This is a higher value than the results obtained for the best MFF approach (threshold of 3) tested with LE-PAROLE (94.78%). However, these cannot be compared, due to the differences on training and testing corpora, as well as the distinct sizes of the tagsets. Both approaches use approximately 80 tags for verbs, however, our work uses a total of 375 tags, while Branco & Nunes use 148 tags. Additionally, they tested the system with only 3,153 verb forms, while our system was tested with 38,927 verb forms.

| # Verbs correctly identified | # Verbs with inflection ambiguity | # Verbs with inflection and lemma ambiguity |
|---|---|---|
| 38,279 | 18,479 | 508 |

**Table 5.1:** Number of verbs correctly classified by MARv4 with no verb disambiguation rules and their ambiguity classes.

With respect to IFtag+lemma measure, the authors did not mention any information.

### 5.5.3 ML disambiguators

As in the MFF approach 5.5.2, the rule-based module, discarding the rules which disambiguate verbal inflection features, is combined with the ML methods in MARv4. For this set of rules, Table 5.1 shows that from the total of 38,927 verb forms in the evaluation *corpus*, 38,279 are correctly identified by MARv4, from which 48.27% have inflection features ambiguity and, 1.37% are also ambiguous in the lemma.

In the next experiments, the sequence of disambiguators developed in MARv4 is the same as the one on MFF experiments, presented in Section 5.8. So, the ML methods for verbal inflection disambiguation will be executed after lemma disambiguation as some verb forms have their own ML disambiguator.

There are five inflection features in the tag of a verb to disambiguate, however, the experiments do not consider specific models for gender disambiguation, as almost all the ambiguity cases regarding this inflection feature are solved by the disambiguation of the other inflection features.

When the disambiguation is a sequential process (as explained in Section 4.3), the tag depends on the order in which inflection features are disambiguated. This section presents the results of ML disambiguators, considering different sequences of inflection features in the disambiguation process.

The first experiments used a ME classifier per inflection feature. All the possible sequences were tested and Figure 5.9 displays the best-achieved results, considering a disambiguation starting with each one of the inflection features. In this figure *m* corresponds to mood, *t* to tense, *p* to person and *n* to number, and the sequence of these letters represents the order of the disambiguation. The models are separated by the underscore symbol.

The highest precision is obtained by the disambiguator that prioritizes the mood inflection feature, achieving the value of 94.52% for only inflection features (IFtag) and 94.06% on the features together with lemma (IFtag+lemma). This model achieves a precision of 2.85% above of the baseline for the first indicator and 2.81% for the second measure.

The disambiguator that starts by number feature presented the lowest results, even slightly below the baseline. It can be also seen that the precision of the IFtag+lemma indicator is consistently lower than the results for the IFtag (without lemma). This is an expected result, since adding the lemma works as an added condition and because, on one hand not all the verbs with lemma ambiguity are disambiguated

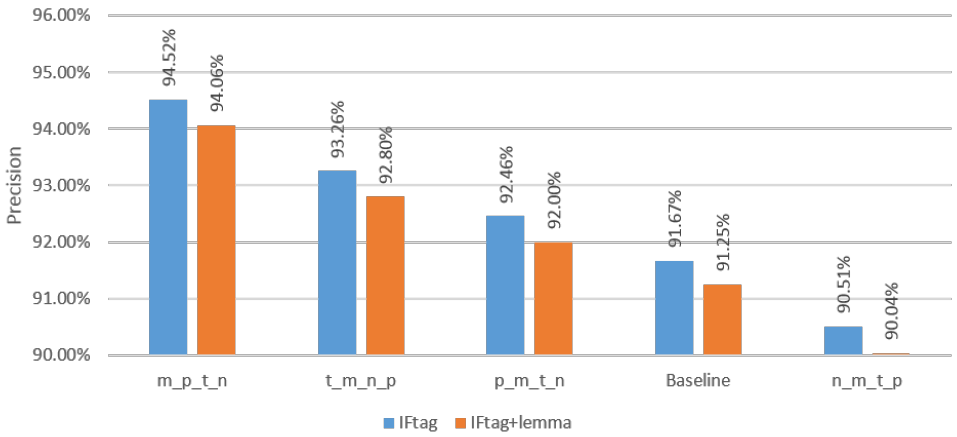


**Figure 5.9:** Results of no verbal disambiguation rules combined with ME method performing as a sequential disambiguator, using four models, which are separated by _ symbol. The letters correspond to each inflection feature.

on the ML disambiguator; and, on the other hand, the ML disambiguator can also classify incorrectly some instances.

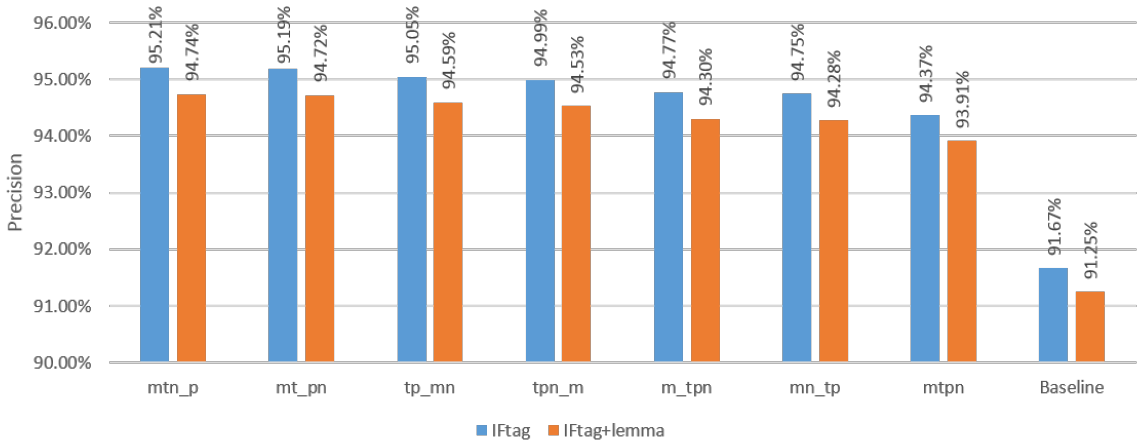


**Figure 5.10:** Results of the no verbal disambiguation rules scenario combined with ME method performing as a sequential disambiguator, using composite models constituted by the inflection features represented by the letters, which are separated by _ (underscore) symbol.

Instead of single models, we also tested all the possible sequences of models composed of several inflection features, instead of single models. The best seven models are shown in Figure 5.10, where the letters represent the inflection features and the models are separated by the underscore symbol. The best ML disambiguator is the one which starts by disambiguating the combination of mood, tense and number inflection features, concluding the process with person disambiguation ($mtn\_p$ in Figure 5.10). This disambiguator yields a precision of 95.21% for IFtag and 94.74% for the IFtag+lemma.

The ML that selects first the most adequate class for mood and tense together, followed by both person and number achieved very similar results when compared with the last mentioned classifier. This configuration obtains a precision of 95.19% for the first indicator and 94.72% for the second indicator. The same figure also shows that, for the disambiguators using composed models, performing a disambiguation with all the features on a single step ($mtpn$ in the figure) is the approach which obtained the lowest precision. All the disambiguators present higher precision than the baseline.

Figure 5.11 shows the results of the sequential disambiguators, using the single models presented before, as well as the results of the sequential disambiguators within the composed models. It is possible to observe that almost all composed models achieved higher precision that the single models. However, the sequential disambiguator with four single models $m\_p\_t\_n$, which prioritizes mood inflection feature achieved better results that the disambiguation on a single step ($mtpn$). The MFF approach achieved a precision higher than all the sequential disambiguators with single models. However, the best ML approach presented is better than MFF technique, presenting a gain of 0.43% for IFtag. Contrasting with the baseline, the gain is 3.54% for IFtag.

### 5.5.4 Naive Bayes experiments

After the tests performed with ME disambiguator, the Naive Bayes approach was tested with the best two single models and for the seven best-composed models, which obtained the highest precision with the ME approach.

Figures 5.12 and 5.13 present the result for both classifiers. The first figure compares the precision obtained for IFtag measure and the second the results for IFtag+lemma. The Naive Bayes approach consistently presents a lower precision, when compared with ME, the differences varying in a range between 0.86% to 2.63% for the IFtag indicator. However, in the Figure 5.12 shows that the disambiguator with the highest precision for ME (95.21%) is not the same as for Naive Bayes (95.05%). Nevertheless, for the disambiguation of IFtag only, the best performing ME disambiguator obtained a gain of 1.7% when compared with the Naive Bayes approach that achieved the best precision.

### 5.5.5 Sequential disambiguation of lemma and inflection features

In the problem of verb disambiguation, there are some verb forms which have several possible lemmas and inflection features simultaneously, as mentioned in Section 1.2. This section analyse and compare the better performing strategy to disambiguate ambiguous verb forms in lemma and inflection features: whether to disambiguate first the lemma or the inflection features. The disambiguators which obtained the highest precision in the previous section (ML $mtn\_p$, ML $mt\_pn$ and MFF) were also tested disambiguating the verbal lemma after the tag.

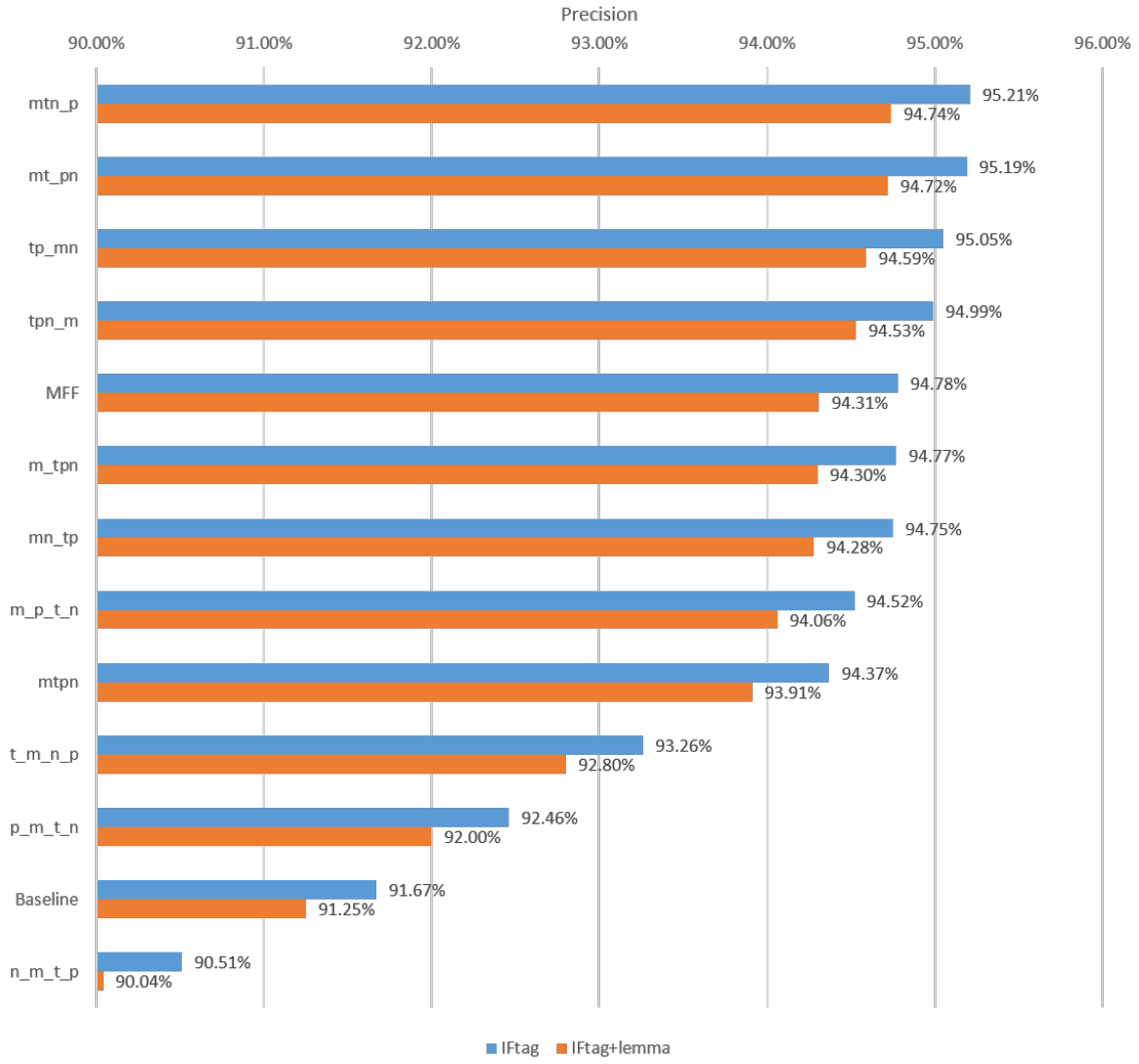


**Figure 5.11:** Results of no verbal disambiguation rules combined with ME classifier with the best single models and composed models for IFtag and IFtag+lemma. The results of the MFF with threshold of 3 and the baseline are also present.

Figure 5.14 shows the precision for only IFtag measure. Both of ML approaches are slightly better when the lemma disambiguation occurs in the first place, however, the differences are not relevant. For the MFF solution, the precision values are also slightly better, when the disambiguation sequence starts by disambiguating verbal lemmas.

When looking at the precision of IF tag together with the lemma, shown in 5.15, the order of which disambiguator performs the disambiguation, either starting by lemma or inflection features, is not significative. A possible justification is that the number of verb forms which MARv4 performs lemma disam-

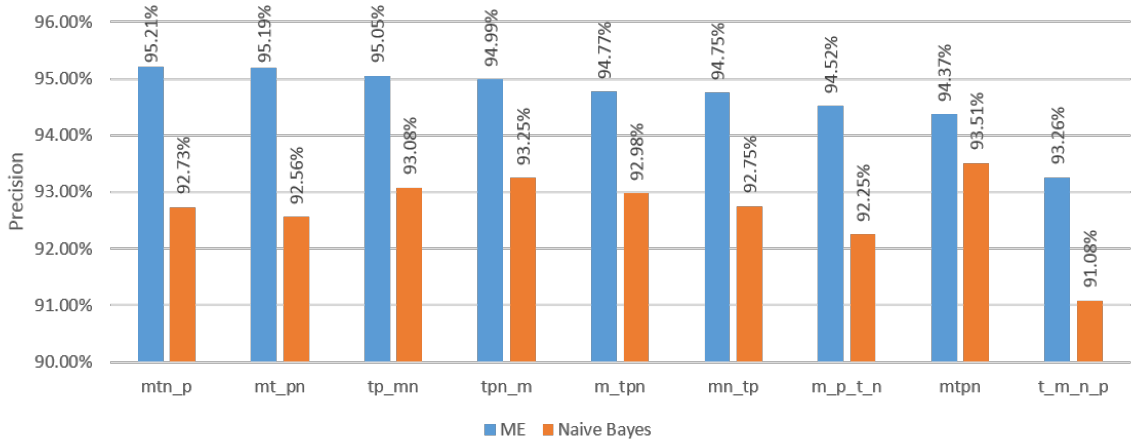


**Figure 5.12:** Results of the no disambiguation rules scenario combined with the best disambiguators achieved with ME method and the respective results for Naive Bayes for the same tests. The results regard only precision on the IFtag assignment for verb forms.

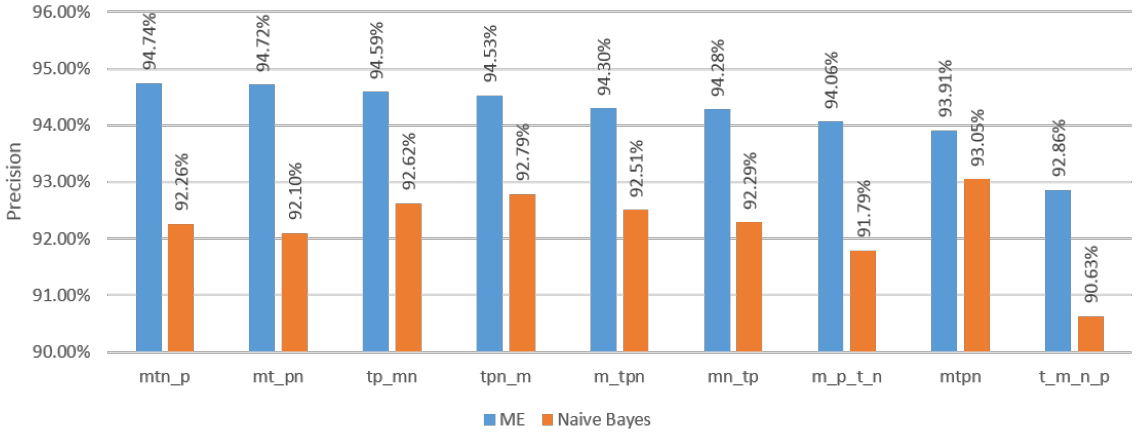


**Figure 5.13:** Results of the no disambiguation rules combined the best disambiguators achieved with ME method and the respective results for Naive Bayes for the same tests. The results regard precision on the IFtag+lemma assignment for verb forms.

biguation appears only 508 times in the testing corpora (as presented in Table 5.1), compared to the 18,479 verb forms with inflection ambiguity.

Therefore, the ML approach represented by $mtn\_p$, combined with verbal lemma disambiguation executed before of verbal inflection features disambiguation, is the best approach to realize feature selection, as it achieved the highest precision for both indicators IFtag and IFtag+lemma.

### 5.5.6 Feature Selection

As shown in Figures 5.14 and 5.15 the classifier that yields the best precision is the ML $mtn\_p$ disambiguator. In this classifier the verbal lemma is disambiguated first, followed by the verbal inflection

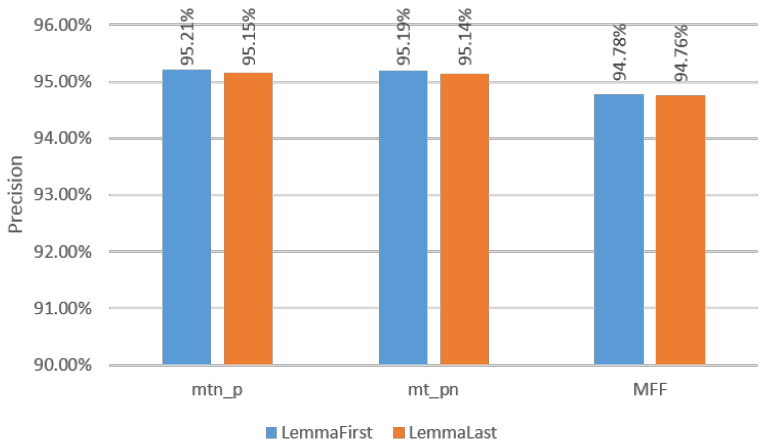


**Figure 5.14:** Results of the best two performing ML approaches and the MFF method for the IFtag indicator: comparison between disambiguating the verbal lemma before the tag (LemmaFirst) or after the tag (LemmaLast).

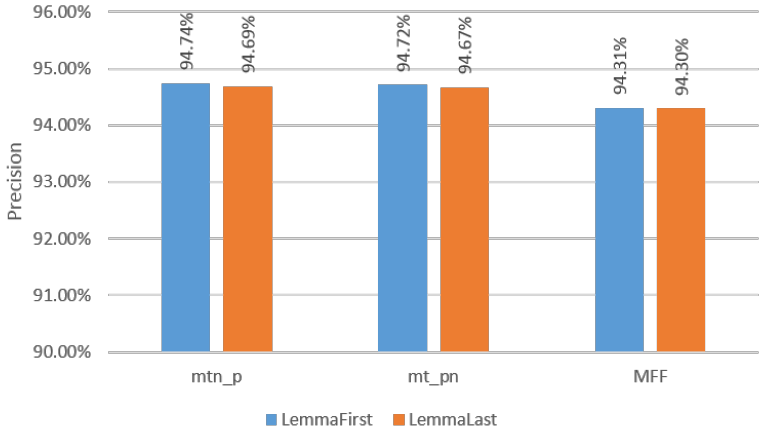


**Figure 5.15:** Results of the best two performing ML approaches and the MFF method for the IFtag+lemma indicator: comparison between disambiguating the verbal lemma before the tag (LemmaFirst) or after the tag (LemmaLast).

disambiguation. Therefore, the impact of the features in the precision was tested for this ML disambiguator.

As described in Table 4.1 several features are related. Therefore, each set of related features were removed at a time. When the precision of IFtag and IFtag+lemma was lower than the previous set of features tested, the removed set is maintained in the system and in the next iteration another set is removed. However if the precision improves without a set of features, in the next iteration it will be not considered.

The difference in the precision among the experiments was not significant, as the variations are always lower than 0.7%, compared with the approach with all the features (in Figure 5.16). Figure 5.16

shows that removing features `VERBFINITENUM, VERBFINITEPER, PPREFLEX, PPREFLEXNUM` and `PPREFLEXPER` (see Section 4.3.3) obtained the highest precision.

The results of lemma disambiguation achieved per verb form, with the best disambiguator (ME $mtn\_p$ without the features above-mentioned), can be seen in Appendix B.

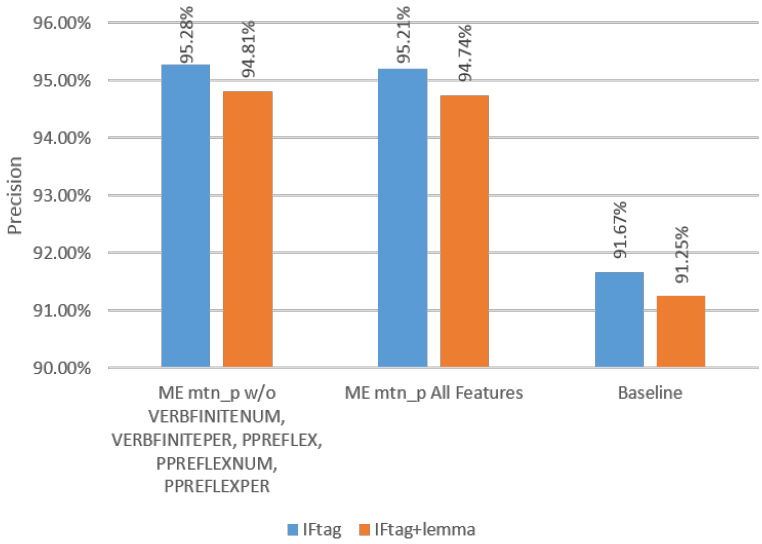


**Figure 5.16:** Results of the best ML approach using all the features, contrasting with the best set of features.

### 5.5.7 Comparison of Several Experiments

Figure 5.17 shows the best results achieved for several experiments. For the ME classifier, the Figure 5.17 presents the approaches which achieve higher precision using a sequence of single models and a sequence of models composed by a combination of verbal inflection features. The sequence of composed models, illustrated as $ME\ mtn\_p\ w/Feature\ Selection$ in the Figure achieved better results than the ME approach using single models ($ME\ m\_p\_t\_n$ in the Figure). Note that this is also the method which obtained the highest global precision, either for IFtag or IFtag+lemma. The result for the best ME classifier was improved over time with feature selection and executing the disambiguation of verbal inflection after the verbal lemmas are disambiguated, obtained a final precision of 95.28% for IFtag and 94.81% for IFtag+lemma. All rules were also combined with the best ME approach $mtn\_p$ with feature selection (precision of 94.72% for IFtag and 94.27% for IFtag+lemma), however, it presents lower results when combined with no verbal disambiguation rules (precision of 95.28% for IFtag and 94.81% for IFtag+lemma).

The best results of Naive Bayes classifier are shown in the Figure 5.17, where both methods (using single models or composed models) presented a precision below the best performing ME classifier.

Even using composed models, Naive Bayes achieved a precision lower than the best ME classifier with single models. Nevertheless, Naive Bayes presents slightly higher results than the baseline.

The MFF achieved a precision of 94.78% for IFtag and 94.31% for IFtag+lemma, considering a threshold of 3, as detailed in Section 5.5.2, and executing the disambiguation as a sequential process, disambiguating by the verbal lemmas first and then selecting the best verbal inflection tag. These results are above the baseline, however, they are lower than the best ME approach, as presented at the beginning of this section.

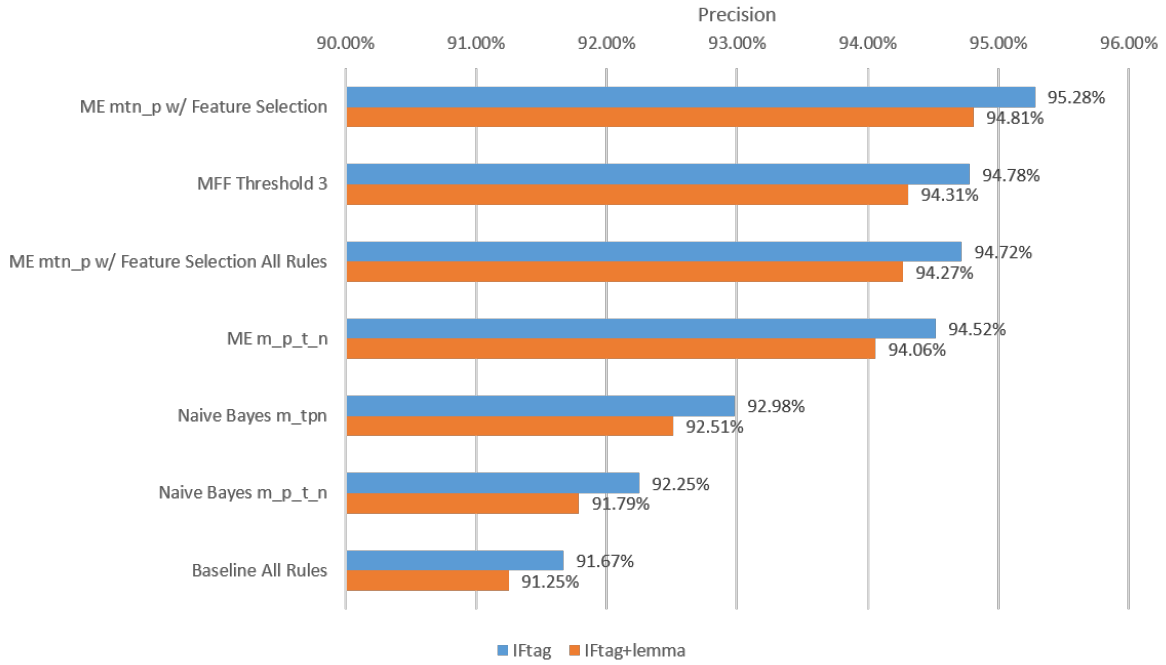


**Figure 5.17:** Results of the best approaches for each one of the classifiers.

# 6

# Conclusions

## Contents

This dissertation addressed the task of full disambiguating of verbal tags. This task consists in assigning the most adequate tag for a verb, considering all its inflection features, namely mood, tense, person, number, and gender. This work was integrated in MARv4 [21], a statistical disambiguator of STRING [18].

Some improvements in other tasks already performed by MARv4 were also a concern in this project. This work has fixed, for instance, some problems regarding category and subcategory disambiguation. Other contributions were the improvement of ML approaches in lemma disambiguation for verbs and in case disambiguation for pronouns, mainly in the feature extraction process. Also, were added 10 new, frequently occurring, verb forms, for which MARv4 is now able to disambiguate their lemma.

To perform the disambiguation of verbal inflection features, the implementation of the ME algorithm was adjusted and a new ML technique was implemented on MARv4, the Naive Bayes approach.

Additionally, a baseline was established, considering all the disambiguation rules combined with the base of MARv4, without any extra disambiguator for dealing with verbal inflection disambiguation. This baseline approach achieved a precision of 91.67% for inflection features only and 91.25% for inflection features together with lemma.

As ME technique proved to be a good approach for lemma and case disambiguation, the majority tests were performed on ME and then compared with Naive Bayes approach. Firstly, experiments with ME algorithm were carried out to determine the best sequence of inflection features, when performing this disambiguation iteratively. The sequence of disambiguating mood, tense and number together, followed by person inflection feature ($mtn\_p$) achieved the highest precision, obtaining a value of 95.21% for verbal inflection features in the verb tag. The Naive Bayes approach was tested with the models that achieved better results with the ME approach. The Naive Bayes always performed below the ME. In the best sequence, codified by $mtp\_n$, with the ME approach, the influence of lemma disambiguation for verbs on the disambiguation of inflection features and vice-versa was also tested, varying the order in which each disambiguation step is performed in MARv4. Irrespective of when the lemma disambiguation was carried out firstly or the lastly in the sequence, the differences in the precision were not conclusive, through the lemma first approach consistently outperformed, even if only slightly the lemma last approach.

Feature selection was also performed on the best model, however, the differences on precision were not significative. Nevertheless, the highest precision achieved was 95.28% for verbal inflection tag only and 94.81% for verbal inflection plus the lemma in the tag, removing the set of features referred in Section 5.5.6.

Therefore, the best ML technique for a full disambiguation of the verb tag achieved a precision of 95.28% for verbal inflection and 94.81% for the first indicator plus the verbal lemma (3.56% and 3.61% respectively, above the results reached by the baseline. With respect to MFF, the best ML technique

reached a precision of 0.5% above that method with a threshold of 3, for both indicators: verbal inflection tag only, and verbal inflection plus the verbal lemma. This work on verbal inflection disambiguation for European Portuguese achieves slightly lower results when compared with the results reported in the previous work, developed by Branco & Nunes [6]. The authors report a precision of 96.51% with a threshold of 1. The best precision achieved in this work was 95.28% based on ME method, contrasting with the version of MFF implemented in the system, which achieves 94.78% of precision with a threshold of 3. However, these works should not be compared directly as the *corpora* used and the size of the tagsets are different. Nevertheless, discarding the rules of verb disambiguation on the verbal inflection disambiguation, the category and subcategory disambiguation present a precision slightly lower than the baseline, as well as case disambiguation of pronouns. In future work some improvements to deal with this issue will be presented.

## 6.1 Contributions

This dissertation contributed to the implementation of a verbal inflection disambiguating system for European Portuguese, using ML techniques. Other contributions are a study of the influence of verbal lemma disambiguation on verbal inflection disambiguation and vice-versa.

Another contribution of this work is the improvement of the MARv4 module of the STRING chain, solving some issues on category and subcategory disambiguation, pronoun disambiguation and, verbal lemma disambiguation. The verbal lemma disambiguation was also extended with 10 new verb forms. This work also addressed the integration of the verbal inflection disambiguator on STRING chain.

A paper presenting a solution to improve the POS disambiguation of personal pronouns in Portuguese was written to be submitted to an international journal.

## 6.2 Future Work

This section presents several ideas that can be implemented in the future, to improve the work developed so far.

The set of rules without any disambiguation rule for the verb inflection features disambiguation was combined with ML techniques. Instead of discarding all these rules at once, one should test the impact of reintroducing each rule at a time.

In the multiple experiments that were carried out, the case pronoun disambiguation always precedes the task of assigning the best POS tag for a verb. Thus, it can be experimented whether the task of choosing the adequate case for pronouns after disambiguating the verb tags can improve the results.

The feature selection regarding verbal inflection disambiguation can be improved by removing only a feature at a time, instead of a set of related features as performed in this work. Consequently, the impact of each feature on the classifier could be observed with more detail. Also related to feature selection, when disambiguating the full tag for verbs with a sequence of several trained models, determining the best features for each model is a research topic that can still be explored

Implementing other ML methods on MARv4 could also be an improvement to the system, in order to analyse their suitability for the task of full tag disambiguation of verb forms: for instance, SVM methods [13], as the state-of-the-art review suggests, or even Condition Random Fields (CRFs) [28].

It would also be interesting to test the ML techniques and the MFF approach with a different evaluation *corpus*, providing evaluation with a context different from the one used to train the models, so as to see the difference in precision between the solutions, and their portability.

# Bibliography


[1] AIT-MOKHTAR, S., CHANOD, J.-P., AND ROUX, C. Robustness Beyond Shallowness: Incremental Deep Parsing. *Natural Language Engineering 8*, 3 (2002), 121–144.

[2] BAPTISTA, J. ViPEr: A Lexicon-Grammar of European Portuguese Verbs. In *Proceedings of the 31st International Conference on Lexis and Grammar* (Nové Hrady, Czech Republic, 2012), J. Radimsky, Ed., Università degli Studi di Salerno (Italy)/University of South Bohemia, pp. 10–16.

[3] BARRETO, F., BRANCO, A., FERREIRA, E., MENDES, A., NASCIMENTO, M. F., NUNES, F., AND SILVA, J. Open Resources and Tools for the Shallow Processing of Portuguese: the TagShare project. In *Proceedings of The Fifth International Conference on Language Resources and Evaluation (LREC)* (Genoa, Italy, 2006), pp. 1438–1443.

[4] BERGER, A., PIETRA, V., AND PIETRA, D. A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics 22*, 1 (1996), 39–71.

[5] BRANCO, A., COSTA, F., AND NUNES, F. The processing of verbal inflection ambiguity: Characterization of the problem space. *Actas do XXI Encontro Anual da Associação Portuguesa de Linguística* (2007), 157–168.

[6] BRANCO, A., AND NUNES, F. Verb Analysis in a Highly Inflective Language with an MFF Algorithm. In *Proceedings of the 10th International Conference on Computational Processing of the Portuguese Language (PROPOR)* (Berlin, Heidelberg, 2012), Springer-Verlag, pp. 1–11.

[7] BRANTS, T. TnT: A Statistical Part-of-speech Tagger. In *Proceedings of the Sixth Conference on Applied Natural Language Processing* (Stroudsburg, USA, 2000), Association for Computational Linguistics, pp. 224–231.

[8] DAUMÉ, H. Notes on CG and LM-BFGS Optimization of Logistic Regression, 2004. http://hal3.name/megam/, accessed June 2016.

[9] DE HOLANDA MAIA, M., AND XEXÉO, G. Part-of-speech Tagging of Portuguese Using Hidden Markov Models with Character Language Model Emissions. In *Proceedings of the 8th Brazilian Symposium in Information and Human Language Technology* (Cuiabá, Brazil, 2011), pp. 159–163.

[10] DINIZ, C., MAMEDE, N., AND PEREIRA, J. RuDriCo2 - A Faster Disambiguator and Segmentation Modifier. In *II Simpósio de Informática (INForum)* (Universidade do Minho, Portugal, 2010), pp. 573–584.

[11] DO NASCIMENTO, M., VELOSO, R., MARRAFA, P., PEREIRA, L., RIBEIRO, R., AND WITTMANN, L. LE-PAROLE: do Corpus à Modelização da Informação Lexical num Sistema-Multifunção. *Actas do XIII Encontro Nacional da Associação Portuguesa de Linguística 2* (1998), 115–134.

[12] FORNEY, G. The Viterbi Algorithm. *Proceedings of the IEEE 61*, 3 (1973), 268–278.

[13] GIMÉNEZ, J., AND MÀRQUEZ, L. SVMTool: A general POS tagger generator based on Support Vector Machines. In *In Proceedings of the 4th International Conference on Language Resources and Evaluation* (Lisbon, Portugal, 2004), pp. 43–46.

[14] GRAÇA, J. A., GANCHEV, K., COHEUR, L., PEREIRA, F., AND TASKAR, B. Controlling Complexity in Part-of-speech Induction. *Journal of Artificial Intelligence Research 41*, 2 (2011), 527–551.

[15] HÖFLER, S., AND PIOTROWSKI, M. Building Corpora for the Philological Study of Swiss Legal Texts. *Journal for Language Technology and Computational Linguistics 26*, 2 (2011), 77–89.

[16] KINOSHITA, J., DO NASCIMENTO SALVADOR, L., AND DE MENEZES, C. E. D. CoGrOO: a Brazilian-Portuguese Grammar Checker based on the CETENFOLHA Corpus. In *The Fifth International Conference on Language Resources and Evaluation (LREC)* (Genoa, Italy, 2006), pp. 2190–2193.

[17] LI, S., GRAÇA, J. A. V., AND TASKAR, B. Wiki-ly Supervised Part-of-speech Tagging. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (Stroudsburg, USA, 2012), EMNLP-CoNLL '12, Association for Computational Linguistics, pp. 1389–1398.

[18] MAMEDE, N., BAPTISTA, J., DINIZ, C., AND CABARRÃO, V. STRING: A Hybrid Statistical and Rule-Based Natural Language Processing Chain for Portuguese. In *11th International Conference on Computational Processing of Portuguese (PROPOR)* (Coimbra, Portugal, 2012), vol. Demo Session. http://www.propor2012.org/demos/DemoSTRING.pdf, accessed June 2016.

[19] OOSTDIJK, N., GOEDERTIER, W., VAN EYNDE, F., BOVES, L., MARTENS, J.-P., MOORTGAT, M., AND BAAYEN, R. H. Experiences from the Spoken Dutch Corpus Project. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC)* (Las Palmas, Spain, 2002), pp. 340–347.

[20] POEL, M., STEGEMAN, L., AND OP DEN AKKER, H. A Support Vector Machine Approach to Dutch Part-of-Speech Tagging. In *Advances in Intelligent Data Analysis VII. Proceedings of the 7th International Symposium on Intelligent Data Analysis (IDA)* (London, 2007), M. Berthold, J. Shawe-Taylor, and N. Lavrac, Eds., vol. 4723 of *Lecture Notes in Computer Science*, Springer Verlag, pp. 274–283.

[21] RIBEIRO, R. Anotação Morfossintática Desambiguada do Português. Master's thesis, Instituto Superior Técnico, Universidade de Lisboa, 2003.

[22] RIBEIRO, R., OLIVEIRA, L., AND TRANCOSO, I. Using Morphossyntactic Information in TTS Systems: Comparing Strategies for European Portuguese. In *6th Workshop on Computational Processing of the Portuguese Language (PROPOR)* (Faro, Portugal, 2003), Lecture Notes in Artificial Inteligence, Springer-Verlag, Heidelberg, pp. 143–150.

[23] RISH, I. An empirical study of the Naive Bayes classifier. In *IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence* (Seattle, USA, 2001), vol. 3, IBM New York, pp. 41–46.

[24] ROCHA, P., AND SANTOS, D. CETEMPúblico: Um corpus de grandes dimensões de linguagem jornalística portuguesa. *Actas do V Encontro para o processamento computacional da língua portuguesa escrita e falada (PROPOR)* (2000), 131–140.

[25] SANTOS, D., HABER, R., AFONSO, S., AND BICK, E. Floresta sintá(c)tica: a treebank for Portuguese. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC)* (Las Palmas, Spain, 2002), ELRA, pp. 1698–1703.

[26] SCHMID, H. Improvements In Part-of-Speech Tagging With an Application To German. *Proceedings of the ACL SIGDAT-Workshop* (1995), 47–50.

[27] SEARA, I., PACHECO, F. S., KAFKA, S. G., SEARA JR, R., AND SEARA, R. Morphosyntactic Parser for Brazilian Portuguese: Methodology for Development and Assessment. In *9th International Conference on Computational Processing of Portuguese Language (PROPOR)* (Porto Alegre, Brazil, 2010), pp. 1–6.

[28] SILFVERBERG, M., RUOKOLAINEN, T., LINDÉN, K., AND KURIMO, M. Part-of-Speech Tagging using Conditional Random Fields: Exploiting Sub-Label Dependencies for Improved Accuracy. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL)* (Baltimore, USA, 2014), vol. 2, pp. 259–264.

[29] SUGISAKI, K., AND HÖFLER, S. Verbal Morphosyntactic Disambiguation through Topological Field Recognition in German-Language Law Texts. In *Systems and Frameworks for Computational Mor-*

*phology: Third International Workshop* (Berlin, German, 2013), C. Mahlow and M. Piotrowski, Eds., vol. 380, Springer, pp. 135–146.

[30] TRUSHKINA, J., AND HINRICHS, E. A Hybrid Model for Morpho-Syntactic Annotation of German with a Large Tagset. In *Proceedings of EMNLP 2004* (Barcelona, Spain, July 2004), D. Lin and D. Wu, Eds., Association for Computational Linguistics, pp. 238–245.

[31] TUFIS, D. Tiered Tagging and Combined Language Models Classifiers. In *Proceedings of the Second International Workshop on Text, Speech and Dialogue* (London, UK, 1999), Springer-Verlag, pp. 28–33.

[32] VICENTE, A. LexMan: um Segmentador e Analisador Morfológico com Transdutores. Master's thesis, Instituto Superior Técnico, Universidade de Lisboa, 2013.

[33] VILLAVICENCIO, A., AND VICCARI, R. Avaliando um rotulador estatístico de categorias morfo-sintáticas para língua portuguesa. Master's thesis, Instituto de Informática, Universidade Federal do Rio Grande do Sul, 1995.

[34] ZHANG, H. The Optimality of Naive Bayes. In *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference (FLAIRS)* (Miami Beach, USA, 2004), V. Barr and Z. Markov, Eds., vol. 1, AAAI Press, pp. 1–6.

# Tag Set used in LexMan and RuDriCo2

| TAG FORMATION | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| **Category** | **Subcat** | **Mood** | **Tense** | **Person** | **Number** | **Gender** | **Degree** | **Case** | **Syntatic** | **Semantic** |
| CAT | SCT | MOD | TEN | PER | NUM | GEN | DEG | CAS | SYN | SEM |

| CATEGORY values | | | |
|---|---|---|---|
| Position | LexMan | Meaning | RuDriCo2 |
| 1 | N | Noun | nou |
| 1 | V | Verb | ver |
| 1 | A | Adjective | adj |
| 1 | P | Pronoun | pro |
| 1 | T | Article | art |
| 1 | R | Adverb | adv |
| 1 | S | Preposition | pre |
| 1 | C | Conjunction | con |
| 1 | M | Numeral | num |
| 1 | I | Interjection | int |
| 1 | O | Punctuation | pun |
| 1 | Y | Symbol | sym |

| NOUN subcategory values | | | |
|---|---|---|---|
| Position | LexMan | Meaning | RuDriCo2 |
| 2 | c | common | com |
| 2 | p | proper | prp |

| ARTICLE subcategory values | | | |
|---|---|---|---|
| Position | LexMan | Meaning | RuDriCo2 |
| 2 | d | definite | def |
| 2 | i | indefinite | idf |

| PRONOUN subcategory values | | | |
|---|---|---|---|
| Position | LexMan | Meaning | RuDriCo2 |
| 2 | p | personal | per |
| 2 | d | demonstrative | dem |
| 2 | i | indefinite | idf |
| 2 | o | possessive | pos |
| 2 | t | interrogative | itr |
| 2 | r | relative | rel |

| NUMERAL subcategory values | | | |
|---|---|---|---|
| Position | LexMan | Meaning | RuDriCo2 |
| 2 | c | cardinal | car |
| 2 | o | ordinal | ord |
| 2 | f | fractional | frc |
| 2 | m | multiplicative | mul |
| 2 | r | roman-and-cardinal | roc |
| 2 | z | roman-and-ordinal | roo |
| 2 | e | det-numeral (e.g. *dezena*) | ena |

| CONJUNCTION subcategory values | | | |
|---|---|---|---|
| Position | LexMan | Meaning | RuDriCo2 |
| 2 | c | coordinate | coo |
| 2 | s | subordinate | sub |

| MOOD values | | | |
|---|---|---|---|
| Position | LexMan | Meaning | RuDriCo2 |
| 3 | i | indicative | ind |
| 3 | s | subjunctive | sbj |
| 3 | m | imperative | imp |
| 3 | c | conditional | cnd |
| 3 | n | infinitive | inf |
| 3 | f | inflected infinitive | iif |
| 3 | p | participle | par |
| 3 | g | gerund | ger |

| TENSE values | | | |
|---|---|---|---|
| Position | LexMan | Meaning | RuDriCo2 |
| 4 | p | present [*presente*] | prs |
| 4 | i | imperfective past [*pretérito imperfeito*] | pim |
| 4 | f | future [*futuro*] | fut |
| 4 | s | perfective past [*pretérito perfeito*] | ppe |
| 4 | q | pluperfect past [*pretérito mais-que-perfeito*] | pmp |

| PERSON values | | | |
|---|---|---|---|
| Position | LexMan | Meaning | RuDriCo2 |

| | | | |
|---|---|---|---|
| 5 | 1 | first | 1 |
| 5 | 2 | second | 2 |
| 5 | 3 | third | 3 |

| **NUMBER values** | | | |
|---|---|---|---|
| Position | LexMan | Meaning | RuDriCo2 |
| 6 | s | singular | s |
| 6 | p | plural | p |

| **GENDER values** | | | |
|---|---|---|---|
| Position | LexMan | Meaning | RuDriCo2 |
| 7 | m | masculine | m |
| 7 | f | feminine | f |

| **DEGREE values** | | | |
|---|---|---|---|
| Position | LexMan | Meaning | RuDriCo2 |
| 8 | n | normal (= positive) | nor |
| 8 | c | comparative (adj,adv) | cmp |
| 8 | s | superlative (adj,adv) | sup |
| 8 | a | augmentative (nou,adj) | aum |
| 8 | d | diminutive (nou, adj, adv) | dim |

| **CASE values** | | | |
|---|---|---|---|
| Position | LexMan | Meaning | RuDriCo2 |
| 9 | n | nominative (subject) | nom |
| 9 | a | accusative (direct object) | acc |
| 9 | d | dative (indirect object) | dat |
| 9 | o | oblique (prepositional object) | obl |
| 9 | r | reflexive | ref |

| SYNTATIC values | | | |
|---|---|---|---|
| Position | LexMan | Meaning | RuDriCo2 |
| 10 | c | clitic | cli |
| 10 | s | splited | spl |
| 10 | r | reduced | red |
| 10 | p | prefixed | pfx |
| 10 | q | prefixed and reduced | pfr |
| 10 | x | sufixed | sfx |
| 10 | z | sufixed and reduced | sfr |
| 10 | t | prefixed and sufixed | psx |
| 10 | u | prefixed and sufixed and reduced | psr |

| SEMANTIC values | | | |
|---|---|---|---|
| Position | LexMan | Meaning | RuDriCo2 |
| 11 | v | not in dictionary | nid |
| 11 | f | loan-word | lwr |
| 11 | a | acronym, abbreviation | abb |
| 11 | e | e-mail address | ema |
| 11 | h | http address | htt |
| 11 | i | IP address | ipp |
| 11 | b | biblical reference | bib |
| 11 | l | decreto-lei | leg |
| 11 | p | phone-fax | pho |
| 11 | n | NIB | nib |
| 11 | d | IBAN | iba |
| 11 | c | BIC | bic |
| 11 | g | ISBN | sbn |
| 11 | j | ISSN | isn |
| 11 | m | car plate | mat |
| 11 | o | other | oth |

Fields used in each Category
green = used
white = not used

| Fields | CAT | SCT | MOD | TEN | PER | NUM | GEN | DEG | CAS | SYN | SEM |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Verb | 1 | | 3 | 4 | 5 | 6 | 7 | | | 10 | 11 |
| Noun | 1 | 2 | | | | 6 | 7 | 8 | | 10 | 11 |
| Adjective | 1 | | | | | 6 | 7 | 8 | | 10 | 11 |
| Adverb | 1 | | | | | | | 8 | | 10 | 11 |
| Pronoun | 1 | 2 | | | 5 | 6 | 7 | | 9 | 10 | 11 |
| Article | 1 | 2 | | | | 6 | 7 | | | | 11 |
| Number | 1 | 2 | | | | 6 | 7 | | | | 11 |
| Preposition | 1 | | | | | 6 | 7 | | | 10 | 11 |
| Conjunction | 1 | 2 | | | | | | | | | 11 |
| Interjection | 1 | | | | | | | | | | 11 |
| Punctuation | 1 | | | | | | | | | | |
| Symbol | 1 | | | | | | | | | | |

# B

# Verbal Lemma Disambiguation

| Verb form | Correct Ident. | #Lemma 0 | Prec. Lemma 0 | #Lemma 1 | Prec. Lemma 1 | Prec. Total |
|---|---|---|---|---|---|---|
| *aposta* | 4 | 4 | 100.00% | 0 | 0.00% | 100.00% |
| *aposto* | 2 | 2 | 100.00% | 0 | 0.00% | 100.00% |
| *cobre* | 4 | 0 | 0.00% | 4 | 100.00% | 100.00% |
| *cobrem* | 1 | 0 | 0.00% | 1 | 100.00% | 100.00% |
| *criam* | 0 | 0 | 0.00% | 0 | 0.00% | 0.00% |
| *descendo* | 1 | 1 | 100.00% | 0 | 0.00% | 100.00% |
| *dita* | 1 | 1 | 100.00% | 0 | 0.00% | 100.00% |
| *ditas* | 0 | 0 | 0.00% | 0 | 0.00% | 0.00% |
| *dito* | 15 | 0 | 0.00% | 15 | 100.00% | 100.00% |
| *entrava* | 1 | 1 | 100.00% | 0 | 0.00% | 100.00% |
| *foi* | 855 | 77 | 98.70% | 778 | 85.86% | 87.02% |
| *fomos* | 3 | 1 | 100.00% | 2 | 100.00% | 100.00% |
| *for* | 34 | 2 | 100.00% | 32 | 93.75% | 94.12% |
| *fora* | 37 | 6 | 83.33% | 31 | 90.32% | 89.19% |
| *foram* | 209 | 15 | 80.00% | 194 | 95.88% | 94.74% |
| *forem* | 11 | 0 | 0.00% | 11 | 90.91% | 90.91% |
| *fores* | 1 | 0 | 0.00% | 1 | 100.00% | 100.00% |
| *formos* | 1 | 1 | 100.00% | 0 | 0.00% | 100.00% |
| *fosse* | 84 | 6 | 100.00% | 78 | 85.90% | 86.90% |
| *fossem* | 20 | 6 | 83.33% | 14 | 100.00% | 95.00% |
| *fôssemos* | 3 | 2 | 50.00% | 1 | 100.00% | 66.67% |
| *fosses* | 4 | 0 | 0.00% | 4 | 100.00% | 100.00% |
| *fui* | 28 | 16 | 93.75% | 12 | 83.33% | 89.29% |
| *gere* | 1 | 0 | 0.00% | 0 | 0.00% | 0.00% |
| *lida* | 3 | 1 | 100.00% | 2 | 100.00% | 100.00% |
| *morta* | 0 | 0 | 0.00% | 0 | 0.00% | 0.00% |
| *mortas* | 3 | 3 | 66.67% | 0 | 0.00% | 66.67% |
| *morto* | 2 | 2 | 100.00% | 0 | 0.00% | 100.00% |
| *mortos* | 2 | 2 | 100.00% | 0 | 0.00% | 100.00% |
| *param* | 0 | 0 | 0.00% | 0 | 0.00% | 0.00% |
| *revisto* | 0 | 0 | 0.00% | 0 | 0.00% | 0.00% |
| *sente* | 6 | 6 | 100.00% | 0 | 0.00% | 100.00% |
| *sentem* | 3 | 3 | 100.00% | 0 | 0.00% | 100.00% |
| *sentes* | 3 | 0 | 0.00% | 3 | 66.67% | 66.67% |
| *tende* | 2 | 0 | 0.00% | 2 | 100.00% | 100.00% |
| *tendo* | 67 | 67 | 100.00% | 0 | 0.00% | 100.00% |
| *vendo* | 10 | 10 | 100.00% | 0 | 0.00% | 100.00% |
| *vi* | 22 | 22 | 100.00% | 0 | 0.00% | 100.00% |
| *vimos* | 5 | 2 | 100.00% | 3 | 66.67% | 80.00% |
| *vir* | 48 | 3 | 100.00% | 45 | 95.56% | 95.83% |
| *vira* | 5 | 4 | 100.00% | 1 | 0.00% | 80.00% |
| *virá* | 1 | 1 | 100.00% | 0 | 0.00% | 100.00% |
| *viram* | 12 | 12 | 91.67% | 0 | 0.00% | 91.67% |
| *virmos* | 1 | 1 | 100.00% | 0 | 0.00% | 100.00% |
| *vista* | 2 | 2 | 100.00% | 0 | 0.00% | 100.00% |
| *vistas* | 0 | 0 | 0.00% | 0 | 0.00% | 0.00% |
| *visto* | 32 | 32 | 100.00% | 0 | 0.00% | 100.00% |

**Table B.1:** Results of precision of lemma disambiguation per verb form, achieved by the ME $mtn\_p$ with feature selection.