Chapter XV

UnderRL Tagger[1]: a free software for Under-Resourced Languages POS tagging

UnderRL Tagger: un software libre para etiquetar POS en Under-Resourced Languages

José Luis Pemberty Tamayo & Jorge Mauricio Molina Mejía
Universidad de Antioquia – Colombia

Abstract: This chapter presents a free software program that can be used for POS tagging in a multiplicity of languages that do not have automatic taggers. The program aims to facilitate the work with corpora in these languages through Natural Language Processing. Its operation allows the manual tagging process to be gradually automated thanks to a system that makes it possible to recall and reuse tags, as well as to handle large amounts of text and to generate output files in XML format with tags based on the EAGLES system.

Resumen: En este capítulo se presenta un software libre que puede utilizarse para el etiquetado de POS en una multiplicidad de lenguas que no cuentan con etiquetadores automáticos. El programa busca facilitar el trabajo con corpus en estas lenguas a través de la lingüística computacional. Su funcionamiento permite que el proceso manual de etiquetado se convierta poco a poco en automático gracias a un sistema que permite recordar y reutilizar las etiquetas, de la misma manera en que permite manejar grandes cantidades de textos y generar archivos de salida en formato XML con etiquetas basadas en el sistema EAGLES.

1 UnderRL Tagger is a free software for semi-automatic POS tagging of languages without many linguistic resources, which has been created within the framework of the college work of J. L. Pemberty Tamayo (2020), within the research team Corpus Ex Machina (Facultad de Comunicaciones y Filología, Universidad de Antioquia). The computer program has been patented in 2020 by J. L. Pemberty Tamayo, J. M. Molina Mejía and M. I. Marín Morales (2020).

1. Introduction

One of the most notorious aspects in the research and study of current Linguistics is the use of textual corpora for various purposes, for example: grammatical analysis (Parodi, 2010; Biber & Finegan, 2014; Jones & Waller, 2015), anaphora resolution (Mitkov, 2014; Poesio, Stuckardt & Versley, 2016; Grajales Ramírez & Molina Mejía, 2019), statistical analysis by means of corpora (Beaudouin, 2016; Brezina, 2018; Wallis, 2021), etc. On the other hand, it is possible to observe the way in which a strong relationship has been established with Computational Linguistics (Mitkov, 2004; Wilks, 2010; Molina Mejía, 2021), precisely for the processing, handling, and interpretation of required amounts of data (Zeroual & Lakhouaja, 2018). Within this scenario, written texts play a prominent role, since they lend themselves to computational processes more easily than other forms of language use (Baquero Velásquez, 2010; Parodi, 2010). Such ease has made it possible to standardize different levels of annotation or tagging, which are ways of enriching the information in the text, making the linguistic notions underlying their use patent (McEnery & Hardie, 2011). An example of this is the POS (Part-of-Speech) level, the simplest and most necessary as a first step in the annotation of texts with linguistic information (Parodi, 2010; Straka & Straková, 2017).

The aforementioned process acquires importance when considering the purposes pursued by Corpus Linguistics, because it permits computers to process information to which they would not otherwise have access. In this sense, software products have also been built that, based on different systems of rules or artificial intelligence, can automatically perform, with a high degree of success, common forms of tagging in different languages, generally the most widely spoken ones such as Spanish, English, French, German, among others (Molina Mejía, 2021).

Automation in the case of corpus tagging is of great importance, since the manual work that would be required to annotate a robust corpus of texts is quite expensive in time, effort and human resources, not to say that it can often seem impossible. This situation places languages that do not have the computerized means to be processed efficiently, at a disadvantage; since the need for manual work limits the information that can be taken for an investigation, as well as it can dissuade potential scholars from dedicating themselves to taking them as an object of work. This group is known as Under-Resourced Languages (henceforth URLa) (Krauwer, 2003).

Considering all of the above, this chapter presents "UnderRL Tagger" (Pemberty Tamayo, Molina Mejía & Marín Morales, 2020), a software that aims to help researchers in the process of tagging textual corpora in URLa, based on a system that permits to recall the tags associated with certain words and automating their annotation as much as possi-

ble (Pemberty Tamayo, 2020). It should be noted that the aim of the work is not to achieve fully automatic tagging, but to assist the manual process, as will be seen in the following pages. This program is the result of work done at the level of conception and elaboration of semi-automatic POS tagging systems for Under-Resourced Languages (Pemberty Tamayo, 2020; Pemberty Tamayo & Molina Mejía, 2020; Pemberty Tamayo *et al*., 2023).

2. State of the Art

As mentioned in the previous section, a clear antecedent of the works whose subject is corpus annotation are the computer platforms and computational tools that currently fulfill the task of automatically tagging large amounts of texts in different languages. Some well-known free access tools are TreeTagger[2] (Schmid, 1994) and TagAnt[3] (Anthony, 2015), which could help with the tagging of some different languages at the Part of Speech -POS- level (Weisser, 2018).

Other prominent names are FreeLing[4] (Padró, Collado, Reese, Lloberes & Castellón, 2010) and Stanford Parser[5] (Schuster & Manning, 2016), which allow annotation at different levels of analysis such as parsing (generation of syntactic trees from dependency grammar and immediate constituents, alternatively), recognition of coreferential chains (anaphora and cataphora), elaboration of semantic graphs, analysis of named entities, etc. Regarding FreeLing, it is important to note that this program uses the EAGLES system as a standard for the annotation of the different human languages.

The EAGLES are a series of conventions adopted by different groups in the work with corpora; they were proposed by the "Expert Advisory Group on Language Engineering Standards" (Leech & Wilson, 1996) and consist of a series of regulations in the use of certain codes for the different possible values in the tagging of POS notions. Bearing this in mind, the work presented here also embraces this standardization, its existence being an important antecedent in the definition of the algorithms described later in this chapter.

Within the framework of the creation of a computer system destined to under-resourced languages and minority languages, it is important to start from a standardized morphosyntactic tagging system. In this way, both researchers and specialists in this type

2 TreeTagger is a tool for annotating text with POS and lemma information. More information can be found at the following link: https://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/

3 TagAnt is a freeware POS tagger built on TreeTagger tool. You can download the tool and find more information at the following link: https://www.laurenceanthony.net/software/tagant/

4 Information regarding FreeLing and the possibility of downloading the tool can be found at the following link: http://nlp.lsi.upc.edu/freeling/node/1

5 The Stanford Parser can be viewed and downloaded at the following site: https://nlp.stanford.edu/software/lex-parser.shtml

of language will be able to understand each other. Starting from this premise, it was decided to aim to have the tags proposed by the EAGLES project. This should permit the program to be used by specialists in minority and under-resourced languages in different geographical and linguistic contexts, and the data obtained from research in different languages to be shared globally. It is also worth mentioning different academic works that focus on the computational treatment of URLa; These works are based on approaches as varied as the annotating of specific languages, such as Arabic and Vietnamese (El-Haj, Kruschwitz & Fox, 2015; Le & Besacier, 2009); speech recognition (Besacier, Barnard, Karpov & Schultz, 2014) or corpus collection by obtaining texts from the web (Scannell, 2007). These works share with "UnderRL Tagger" their concern for this group of languages, but they also have the difference that they do not properly deal with automated assistance in manual corpus tagging and their approaches are, in most cases, monolingual.

Unlike these studies, two remarkable computer programs have also been found, since, although they do not mention the concept of URLa in their documentation, they mark more notable antecedents in relation to the objective of this work. These are "FieldWorks Language Explorer" (Moe, 2008) and "Field Linguist's ToolBox" (Buseman & Buseman, 2013), both designed to manage corpora in different languages, mainly with the intention of processing them at the lexicographic level and in order to finally produce a dictionary of the languages worked by each of them (Rogers, 2010).

However, these software programs, given the breadth of their field of application, could hinder the simplest task of obtaining an annotated corpus in each language, in addition to the fact that they also lack a standardization in the field of Corpus Linguistics such as those mentioned in EAGLES. In this sense, they are established as antecedents of this work, but their functionalities are not the same as those of "UnderRL Tagger" (Pemberty Tamayo, 2020).

3. Theoretical Framework

3.1. Computational Linguistics and Natural Language Processing

Computational Linguistics is usually defined as a discipline whose purpose is the construction of computer systems that process linguistic structures and simulate human linguistic capabilities (Moreno Sandoval, 1998, pp. 29-30). This discipline is framed within Applied Linguistics (Moreno Sandoval, 1998; Tordera Yllescas, 2011, Molina Mejía, 2021) and, following the opinion of several authors (Sáiz Noeda, 2002; Tordera Yllescas, 2011), it will be considered in this chapter as a synonym of NLP (Natural Language Processing).

Although many authors agree on this general definition, there are different ways of delimiting the scope of Computational Linguistics. From practical approaches that include all types of computer language processing (Mitkov, 2004, p.15), to more theoretical points of view, which focus on how the simulation of linguistic capacity helps to understand linguistic behaviour of natural languages (Tordera Yllescas, 2011). Considering, in addition, the use or creation of computational models or tools that allow the computational processing of natural languages, which should permit, a fortiori, that the language itself can serve as an input for scientific research and/or formulation of programs that can be applied in life, in society in general, thanks to the analysis of linguistic corpora in context (Molina Mejía, 2021).

In this difference of opinions, intermediate approaches have been found, such as that of Moreno Sandoval (1998), who proposes the following applications: a) systems that try to emulate the human capacity to process natural languages; b) programs to aid writing and textual composition; and c) computer-assisted teaching and linguistic task support systems (pp. 27-29). This last group includes tools for managing and annotating linguistic corpora, i.e., the work presented here. This list of applications can be extended with more current functionalities, following Nerbonne (2007) and Molina Mejía (2021): a) speech recognition; b) speech synthesis; c) data mining; d) automatic completion systems in smartphones; e) management of academic documents and databases; f) conversational systems; g) automatic topic detection; h) automatic summarization; i) automatic document classification, among others.

It is also common to find that Computational Linguistics is understood from its division into theoretical and applied. Theoretical Computational Linguistics deals with the construction of linguistic abstractions that encompass both computer and natural language phenomena, as well as the construction of algorithms that help model and test these abstractions (Nerbonne, 2007, p.3). Applied Computational Linguistics is dedicated to the construction of computer tools to manipulate language for different purposes (Nerbonne, 2007). The delimitation of these applications, as mentioned above, varies depending on the authors, however some may be mentioned: a) automatic translation; b) information retrieval; c) human-machine interfaces; d) text analysis tools; e) lexicographic databases; f) spelling, syntax, and style checkers; and g) educational programs for language teaching (Moreno Sandoval, 1998, pp. 27-29).

3.2. Corpus Linguistics

Corpus Linguistics is defined as a "methodology for languages and language research, which allows empirical **investigations** to be carried out in authentic contexts" (Parodi, 2010, p.15). Considering the empirical and authentic character indicated by this definition, this methodology can be related to the functionalist model of linguistics, which seeks to understand linguistic phenomena in real situations. This model is opposed to the generativist model, which is dedicated to theorizing about phenomena through linguistic intuition (Baquero Velásquez, 2010, p.25; McEnery & Hardie, 2013).

s tasks that fit within Corpus Linguistics, we can include the collection, processing and analysis of large amounts of data representative of the use of the language or languages that are assumed as object of study (Baquero Velásquez, 2010; Bernal Chávez & Hincapié Moreno, 2018; McEnery & Hardie, 2011). There is, moreover, a marked interdisciplinarity in this methodology, as it works both for the investigation of phenomena at any level of the language and to help in meeting the objectives of different fields of Applied Linguistics (Parodi, 2010, p.15).

Given that authenticity, representativeness and interdisciplinarity have been such important aspects in working with corpora; the relationship that can be established between Computational Linguistics and Corpus Linguistics becomes evident, since the former has provided the necessary mechanisms for handling large amounts of data information and its processing by various means (Baquero Velásquez, 2010; Bernal Chávez & Hincapié Moreno, 2018; Parodi, 2010) and, on the other hand, the need for corpora that possess a high level of quality and variety in discourses and textual typologies (Molina Mejía, 2021).

This relationship is even taken for granted nowadays, through authors who go so far as to define a corpus as a series of texts that can be processed by computers (McEnery & Hardie, 2011, p.1). However, this relationship has not always been present, and in previous times, such as the mid-twentieth century (Bernal Chávez & Hincapié Moreno, 2018, p.12) and even the nineteenth century (Baquero Velásquez, 2010), it has been necessary to carry out work with corpora manually. This implied enormous complications, since the more the amount of data with which one works grows, the greater sums of time, money, effort, and human capital are necessary, making some tasks unfeasible (Mitkov, 2001, p.110).

The help of computational means has therefore come to reduce the resources required in these jobs and also the risk of human errors and loss of information. However, not all languages have the appropriate tools to make use of these technologies, which places them at a considerable disadvantage, insofar as it is not possible to carry out work of the same

magnitude with them as with languages that are more accessible to computer processing (Baquero Velásquez, 2010, p.28).

3.2.1. What is a corpus?

The term corpus has already been used in the previous sections and, before continuing, it is necessary to dedicate a few paragraphs to clarify its definition. We will start from the proposal of Bernal Chávez and Hincapié Moreno (2018), for whom a corpus is a set of digital texts that are collected and systemized following linguistic criteria. Note in this definition the importance of computational means with respect to the need for texts to be digital; in addition to this, it is also fundamental the fact that the collection and systematic organization of the corpus is done with respect to these linguistic criteria; this is the main characteristic that distinguishes a corpus from any other collection of texts.

For its part, Parodi (2010) proposes a more specific list of characteristics that can guide us in understanding what a perfect corpus is:

1 Collection of texts in natural environments.
2 Explicitly of the defining features shared by the constituent texts.
3 Final plain digital type format (*.txt) for each text or document.
4 Size, preferably large.
5 Respect for ecological principles.
6 Semi-automatic computational tagging or annotation of a morphosyntactic or other nature for each text.
7 Availability through computational means.
8 Access to complete visualization of the texts that compose it in plain format.
9 Search for principles of proportionality or representativeness (possibly statistical).
10 Livelihood or initial provenance specified.
11 Identification of an organization around themes, types of texts, registers, genres, etc.
12 Record of quantitative data that allows the comparison and possible normalization of figures (p.26).
13 And to comply with all these elements at the same time, but that the importance of each one can vary depending on the specific objectives of each collection of texts (p.27).

In these characteristics, the need for computational processing is also evident, as well as the need to make explicit the features shared by the texts; this may or may not be part of a tagging or annotation, which is also part of the above list. With this in mind, an important part of corpus work is usually the enrichment of textual information with other types of

information that provides clarity about the underlying linguistic notions. This process is known as tagging, and it will be the object to be dealt with in the next section.

3.3. Corpus Annotation

The construction of a corpus is a process that goes through different phases, which include its design, data capture, storage system planning and text processing (Bernal Chávez & Hincapié Moreno, 2018, p.53). Within this last step is a process called annotation.

A clearer definition of corpus annotation can be found in the work of McEnery and Hardie (2011): "[…] is largely the process of providing —in a systematic and accessible form— those analyses which a linguist would, in all likelihood, carry out anyway on whatever data they worked with" (p.13). It is very important to take into account, from this definition, the fact that the data included in the tagging are those that a linguist could extract from the collected texts, that is, the linguistic information that is implicit within the use of language and that it must be made visible in a systematic way so that it can be recognized and processed by computer programs.

To achieve this systematic way of describing the information, specialized languages are used in tagging, which help to assign different types of values to each of the elements of the text, depending on what is to be said about them. Some of these languages are XML (Extensible Markup Language), HTML (HyperText Markup Language) and GML (Generalized Markup Language), as Bernal Chávez and Hincapié Moreno (2018, p.57) explain. JSON (JavaScript Object Notation) language and some standardized formats such as TEI (Text Encoding Initiative) are also used very frequently, according to Molina Mejía (2021). Thus, the result of a tagging process is usually a text in a format different from the original, in which part of its implicit information is made visible.

The information that could be included in corpus annotation can be as wide as the elements that play a role in communication are different and as varied as the objectives that each researcher has when planning the construction of the corpus. In this sense, there is great freedom in choosing what will be explicit in the tags of a corpus. However, in current work it is possible to note that some forms of tagging have become standardized.

Two common types of annotations are the syntactic parsing, which focuses on analysis of the functions that each word fulfils in the syntax of the sentence (Parodi, 2010, p.40) and the POS (Part-of-Speech) tagging, also known, following Mitkov (2004), as morphological or lexical annotation. Although the term part-of-speech refers to something specific, this type of tagging usually presents, in addition to this data, information on gender, number, case, tense, mood, aspect and person (p.225).

There are different approaches to perform this task. For McEnery and Hardie (2011, p.49), a corpus can be tagged manually, automatically or an automatic process followed by a manual review. The application of these methods may vary in their margin of error and in the time and effort to be devoted to tagging, but as will be seen below, their choice depends on how easy it is for a researcher to access automatic tagging methods in a given language.

3.4. Under-Resourced Languages

Considering the aforementioned concepts, the importance of having properly compiled and annotated corpora is evident, as well as the availability of tools for automatic language processing in the studies that can be carried out in a given language (Pemberty Tamayo, 2020). Thus arises the concept of Under-Resourced Languages, which can be defined as the set of languages that do not have the computer resources for their automatic processing, as well as the lexicographic and corpus inputs that would serve as the basis for the construction of these tools (Krauwer, 2003).

A definition can also be found in a series of criteria proposed in the works of Krauwer (2003) and Berment (2004), which propose the tools that a language must have in order to be considered as having a basic level of access to computational linguistics technologies. Languages that lack several of these elements are thus considered to be Under-Resourced Languages:

a Lack of a single writing system or a stable spelling.
b Limited presence on the web.
c Lack of experts in Linguistics.
d Lack of electronic resources for speech and language processing.
e Lack of monolingual corpus.
f Lack of electronic bilingual dictionaries.
g Lack of transcribed oral corpus.
h Lack of pronunciation dictionaries and vocabularies.

As Maxwell & Hughes (2006, p.29) mention, the availability of such tools in a language, coupled with other extralinguistic factors, can greatly influence a researcher's decision to work with it. This means that the lack of tools makes research in some languages less frequent and, therefore, the creation of the same tools could be slow and difficult. The availability of these elements, at the same time, makes different applications of information and communication technologies, such as machine translation or digital dictionaries, available

to speakers of the language. That is why filling the gap in terms of tools for computational processing in these languages is not only an academic interest, but also benefits the communities in which the language is spoken (Pemberty Tamayo, 2020).

Based on all the topics explored in this section, the need for tools for corpus tagging in Under-Resourced Languages is evident. The UnderRL Tagger tool (Pemberty Tamayo *et al.*, 2020) proposes, through Computational Linguistics, a system that allows manual tagging of large amounts of texts in different languages, with the help of the computer, which provides the facility to speed up the process by a significant proportion. This process can also produce content that can be reused to annotate other corpora in the same language and serve as a basis for the creation of applications that allow the fully automatic tagging of texts (Pemberty Tamayo, 2020; Pemberty Tamayo *et al.*, 2023).

4. Methodological Framework

Before describing the methodology through which this software is built, it is necessary to explain some elements that have served to frame it in a standard that facilitates its use in the current environment.

Taking into account that the main objective of the application has been selected as the POS level in tagging, the use of the EAGLES tag system (Leech & Wilson, 1996) was accepted for this purpose, which allows coding information such as grammatical category, gender, number, etc., in a brief way, through different numbers and letters. An example is shown below:

Table 1. Example of EAGLES tags for a Spanish sentence.

I	*BUY*	*BREAD*
PP1CSN0	VMIP1S0	NCMS000

The table above shows how EAGLES tags are used to specify the information for each of the words. However, these series of letters and numbers must be converted into a markup language that can be computationally processed and parsed. To achieve this goal, the program uses the XML language, which allows assigning individual elements within a series of defining characteristics. Thus, in this language the corresponding tag can be assigned to each of the text components. Both the EAGLES tags and the XML language correspond to standards widely used in the corpus tagging environment, so their use guarantees understanding by a wide variety of researchers in the field, as well as easy integration with previous projects or work that may have been carried out.

4.1 Description of the program structure

The UnderRL Tagger software interface consists mainly of a window that can be interacted with to navigate between corpus files, set tags and save or retrieve previous sessions. This window constantly interacts with other files and folders that record everything necessary to make the tagging process as efficient and correct as possible.

One of the folders is used by the system to store the data of the different dictionaries that are created. The dictionary is a file in which the tags that can be reused in a given corpus are stored, so that it is not necessary to re-enter them manually.

Another important location is the folder where the XML files containing the already tagged texts are stored; this folder is automatically created in the same directory as the original corpus texts. In addition, there is also a set of files that record at all times which annotation projects are running and what their progress is; so, it is easy to interrupt the tagging task at any time and come back to it later.

From here, the program can enter all the texts that make up the corpus, which must be in plain text format (*.txt) and UTF-8 encoding, in which the computer will recognize a wide variety of characters. All of them must be stored in a single folder, the address of which will be entered in the application.

Once the texts are available, the software will proceed to go through each of them, as selected by the user, and perform a process that consists of separating the text by words. Once the words have been separated, the main window shows the user each one of them, allowing the user to select more than one when necessary. For each word, the user can select, through several controls, the characteristics of the word to be tagged and the program takes care of representing them according to the EAGLES model. In addition, a space in the interface permits the creation of new tags or the editing of the default ones; in this way it is possible to expand the tagging possibilities according to the needs outside the POS. Finally, once a tag has been established, the user can save it in the final XML file, where it will be arranged with the rest of the text, with its corresponding tag and a unique identifier.

In addition to simply tagging the word, the user can choose to save that tag in the dictionary, so that each time the same word appears in the corpus, it will be automatically tagged without user intervention. This is how this software helps to greatly automate annotation, as it allows human intervention to be reduced to the points where it is really necessary. Each time the tagger encounters a new word, it looks it up in the dictionary before displaying it on the screen, so the same text can go through considerable chunks before requiring human attention.

As a consequence of this procedure, the dictionary can be strengthened as the tagging progresses, permitting for greater automation and also providing a file that can be used to tag other texts in the same language or as a basis for other programs that require knowledge of these notions for language processing.

When a user perceives that the tagging of a word cannot be automated because it may present variations in its tags throughout the corpus, he can simply choose not to save it in the dictionary, so that each time it appears he will be presented in the main window of the interface and will be allowed to choose the tag he considers appropriate for each occasion, as mentioned in Pemberty Tamayo (2020).

5. Analysis of the algorithms

UnderRL Tagger is a software written in Python language that can be used for semi-automatic tagging of POS in Under-Resourced Languages, putting the methods of Natural Language Processing at the service of Corpus Linguistics, and allowing the tagging process to be significantly speeded up by automating several of its stages (Pemberty Tamayo, 2020; Pemberty Tamayo *et al.*, 2023).

When a user correctly enters the address of a folder containing the texts of a corpus, the first actions performed by the program are to verify the existence of the texts and to create the files and folders necessary to store the records involved in the process (Figure 1), as described in the methodological framework.

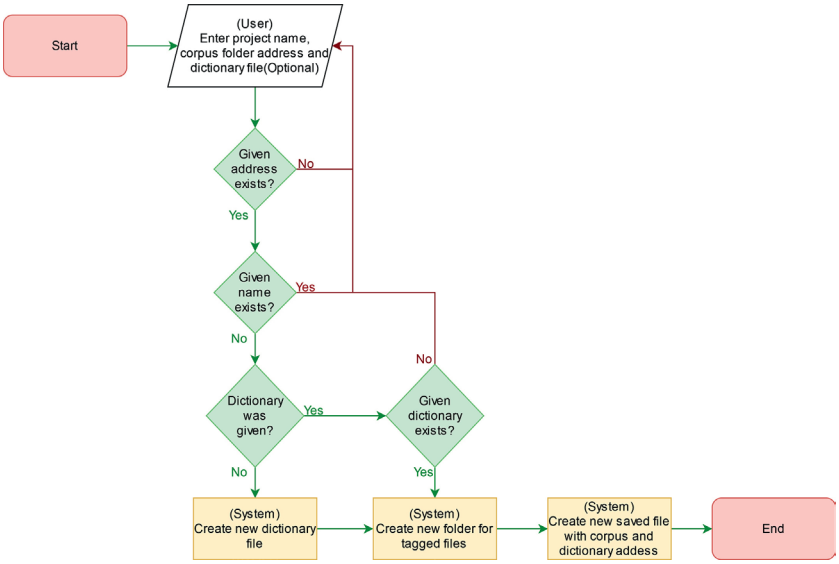

Figure 1. Flowchart: Starting a New Project. Adapted from Pemberty (2020).

All the information that the System stores in addition to the XML tagged texts is in folders that must be in the same directory in which the program is running, and for this purpose files are used that are also in plain text format, so that they can be easily read and modified in case a mistake has been made, for example, by creating an erroneous tag in the dictionary.

Once these files have been prepared, the tool goes on to tag the texts. To exemplify what will happen in each of the steps, we will take here the same sentence that is proposed in the work from which this program arises. This fragment is an example of the Creole language of the islands of San Andres (Colombia) and is shown below along with a brief analysis (Table 2):

Table 2. Description of the "Sentence A" (Pemberty, 2020, p.31).

Sentence A							
Word	*Di*	*bwai*	*gwain*	*da*	*di*	*niu*	*house*
POS	Article	Name	Verb	Preposition	Article	Adjective	Name
Translation	The	boy	goes	to	the	new	house

Before showing the user the texts to be tagged and the diverse options, it is necessary that the text is processed in a specific way. In previous sections it has been said that the text is divided into words and categories are assigned to each of them. In this sense, it is important to specify that the appropriate concept is not that of a word, but that of a token. According to Mitkov (2004), a token is a minimal linguistic unit that can correspond to a word, a number, or a punctuation mark. An important difference between a token and a word is that the latter remains a single element regardless of whether it appears several times in one or in many texts, whereas the former corresponds to a single occurrence, so each of them must be differentiated in relation to the others. The process of dividing a text into its component tokens is called tokenization.

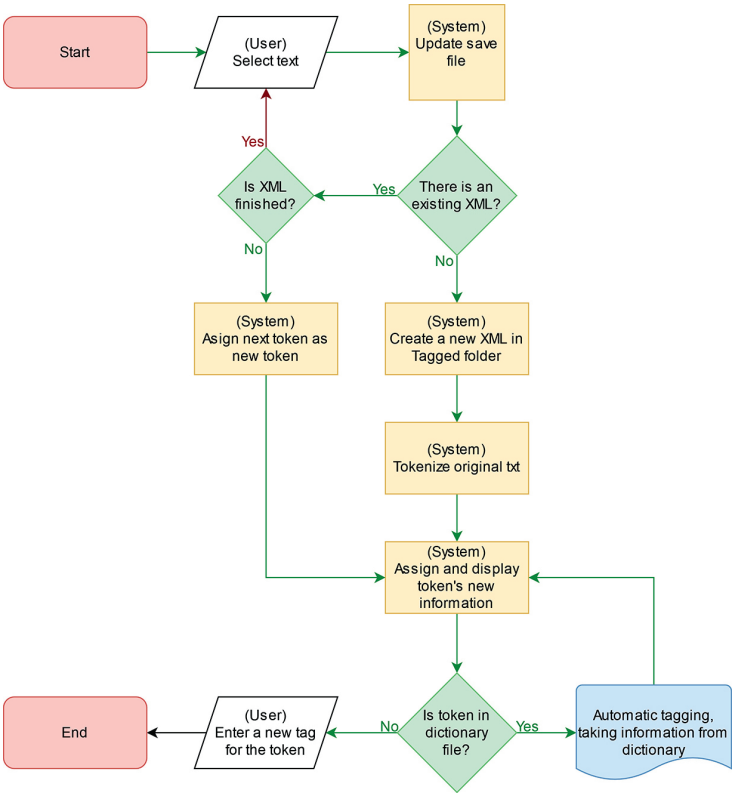

Figure 2. Flowchart: Pre-processing of a selected text. Adapted from Pemberty (2020).

The software checks the file system to see if there is previous information on the same text so that it can be retrieved and continue where the work left off, as well as checking from the first token of the text if there is a set of tags for it in the dictionary, as can be seen in the diagram above. Assuming that this is a new project that has no tags in its dictionary, the result of this process will simply be the tokenized text.

It is also important to note that tokens are usually identified through the blank space between two words; however, there are also many units that are made up of two or more words separated by spaces that would be erroneous to tagged as distinct or non-consecutive tokens. These units are called multi-token words and examples of them can be phrases or some ways of referring to numbers (Mitkov, 2004). To annotate these units, the system offers the possibility of chaining some tokens with others, being able to create a composite unit between one element and the one that follows it.

All the checks seen in Figure 2 are performed automatically by the system, so for the user only a moment passes between selecting a text to tag and the first tokens and controls to set the tags are displayed in the window.

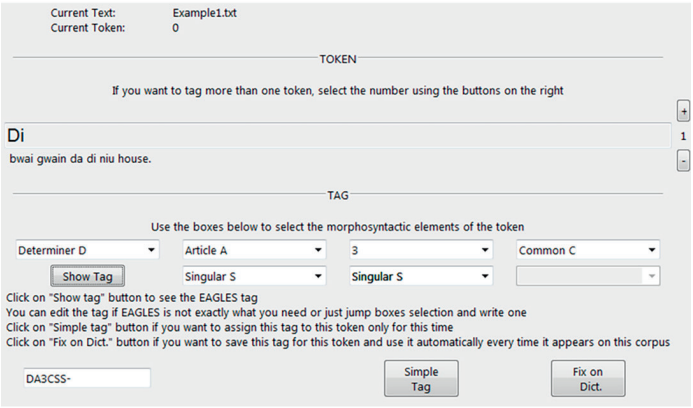

Figure 3. Example of the program window with a tagged unit (Pemberty Tamayo, 2020, p.33).

The program presents the user with the first token of "Sentence A" as well as others that are useful for understanding the context in which each one appears, as shown in Figure 3. Likewise, a series of drop-down lists are enabled for the user that will permit him to choose between distinct categories that could be assigned to the token that is selected. From the various selections, the tag will be created.

The diverse possibilities available to the user vary depending on the first selection to be made, that of the part of speech to be attributed to the token, from which the others are derived. Thus, the amount of information required and its type change when one of these categories is selected.

Once you have selected the appropriate items in the drop-down lists, click on the "Show tag" button, which permits the user to visualize, in the text bar at the bottom, the tag that has been created from the information entered and following the EAGLES system. In the drop-down lists the options are expressed with words commonly used in the field of Linguistics, while the tag only shows its equivalent in the annotation system, as shown in the previous image; in this way, it is not necessary for the user to be perfectly familiar with the EAGLES tags to be able to use them, since the program takes care of establishing which characters are necessary.

The user can already set that tag for that token; however, he be able also to edit it, in case he needs to add additional information of interest for his work. Thus, the tagger per-

mits researchers to create their own tags based on EAGLES or completely new ones, so it could be used not only for URLs, but also in other languages to tag phenomena outside the POS level. This flexibility let the user to work according to the theory or linguistic approach he prefers or needs.

There are also two options to fix the tag and bring it definitively to the output XML file. The first is "Simple Tag", which takes whatever is on the bar where the tag appears and fixes it in the output file associated with that particular token and its ID number.

On the other hand, there is a button called "Fix on Dict". It permits to fix what is written in the tag bar in the dictionary file associated to the selected token; besides that, it performs the procedure of fixing that occurrence of the token in the XML file.

This second option should only be applied when there is certainty that the same tag could be used on all occasions when the same word or combination of words occurs in the token. This can easily be applied to articles, punctuation marks, prepositions, or adverbs, and even to most nouns, adjectives and verbs. This feeds the dictionary, which will be used to automatically tag tokens that match the information it contains. For cases where the tag may vary, the first option will be used, as the absence of that tag in the dictionary will always prompt the user to manually select the appropriate categories. An example dictionary file is shown below:

```
entry_ . **** Fp
entry_ bwai ***** NCMS ---
entry_ di ***** DA-CNS-
entry_ house ***** NCFS---
entry_ niu ***** AQ-CS--
```

Figure 4. Tokens and dictionary entries (Pemberty Tamayo, 2020, p.38).

As shown in Figure 4, this file consists of several lines of text that associate each token with the tag that has been assigned to it. The characters found at the beginning and in the middle of each line are used by the system to differentiate these two elements. The dictionary lookup consists of going through this set of alphabetically ordered lines and taking from them the tag if a match is found, and then taking it to the output file.

By constantly repeating the process of feeding the dictionary with new tokens and tags and allowing the tagger to automatically find and fix as many word occurrences as possible, a significant reduction in the effort required to have a fully XML tagged corpus is achieved.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<text name="Ejemplo1.txt">

<token form="Di" tag="DA3CNS" id="t.0.1"/>
<token form="bwain" tag="NCMS--" id="t.1.1"/>
<token form="gwain" tag="VMIP3SC" id="t.2.1"/>
<token form="da" tag="SP----" id="t.3.1"/>
<token form="di" tag="DA3CNS" id="t.4.1"/>
<token form="niu" tag="AQ-FS-S" id="t.5.1"/>
<token form="house" tag="NCFS--" id="t.6.1"/>
<token form="." tag="Fp" id="t.7.1"/>
</text>
```

Figure 5. Final XML example.

Finally, Figure 5 illustrates what "Sentence A" tagged with the UnderRL Tagger system would look like in your output file. The XML file has an identification of the text in question and all the tokens that make it up. For each of these tokens, the form information is available, which is the exact way it appears in the text; tag, which is the annotation that was established for it and an ID, which is a number that identifies it and differentiates it from all other tokens in the text. This ID is composed of the letter "t", an integer that refers to the position of the token in the text and another integer that refers to the number of words that make up the token, which varies in the case of multi-token words.

6. Conclusions and Perspectives

During this chapter we have seen how it is possible to use Natural Language Processing applications in corpus tagging in languages that do not yet have access to automatic annotation tools, making it possible that, through diverse processes, to achieve a part of what would be enormously expensive if executed completely manually.

The UnderRL Tagger software (Pemberty Tamayo *et al.*, 2020), the tool described in the previous pages, aims to bring URLa closer to information and communication technologies, as well as to facilitate to have them as an object of investigation. For all these reasons, as we have seen in the theoretical framework of this chapter, the existence of computer tools capable of processing and tagging corpora in these languages is of utmost importance.

Thus, through a window-based interface and simple controls, UnderRL Tagger enables a highly computer-assisted and automated manual handling tagging process, offering users the possibility to adhere to international standards in the field of Corpus Linguistics, choose their own tagging system and even annotate outside the POS with any other desired phenomena. Similarly, it allows the management of dictionary files that can be used in the future to further tag texts in the same language or share them with other researchers. Finally, it is important

to note that this software is freely available and can be found in the repository of the main author of this work: https://github.com/jluispemberty/UnderRlTagger.

References

Anthony, L. (2015). *TagAnt (Version 1.2. 0)[Computer Software]*. Waseda University. http://www.laurenceanthony.net/software/tagant/

Baquero, J. M. (2010). *Lingüística computacional aplicada*. Universidad Nacional de Colombia.

Beaudouin, V. (2016). Statistical Analysis of Textual Data: Benzécri and the French School of Data Analysis. *Glottometrics*, 33.

Berment, V. (2004). *Méthodes pour informatiser les langues et les groupes de langues " peu dotées "* [PhD Thesis, Université Joseph-Fourier - Grenoble I]. https://tel.archives-ouvertes.fr/tel-00006313

Bernal, J., & Hincapié, D. (2018). *Lingüística de corpus*. Instituto Caro y Cuervo.

Besacier, L., Barnard, E., Karpov, A., & Schultz, T. (2014). Automatic speech recognition for under-resourced languages: A survey. *Speech Communication*, *56*, 85-100.

Biber, D., & Finegan, E. (2014). On the Exploitation of Computerized Corpora in Variation Studies. In *English Corpus Linguistics* (pp. 216-232). Routledge.

Brezina, V. (2018). *Statistics in corpus linguistics: A practical guide*. Cambridge University Press.

Buseman, K., & Buseman, A. (2013). *Field Linguist's ToolBox (Version 1.6.1)*. SIL International. https://software.sil.org/toolbox/

El-Haj, M., Kruschwitz, U., & Fox, C. (2015). Creating language resources for under-resourced languages: Methodologies, and experiments with Arabic. *Language Resources and Evaluation*, *49*(3), 549-580.

Grajales Ramírez, A. & Molina Mejía, J. (2019). Problemática actual del procesamiento computacional anafórico: el caso de FreeLing 4.1. *Lenguaje*, *47*(2S), 537-568.

Jones, C. & Waller, D. (2015). *Corpus Linguistics for Grammar: A Guide for Research*. Routledge.

Krauwer, S. (2003). The basic language resource kit (BLARK) as the first milestone for the language resources roadmap. In *Proceedings of SPECOM 2003* (pp. 8-15).

Le, V.-B., & Besacier, L. (2009). Automatic speech recognition for under-resourced languages: Application to Vietnamese language. *IEEE Transactions on Audio, Speech, and Language Processing*, *17*(8), 1471-1482.

Leech, G., & Wilson, A. (1996). *EAGLES recommendations for the morphosyntactic annotation of corpora*. Istituto di Linguistica Computazionale http://www.ilc.cnr.it/EAGLES96/annotate/node1.html

Maxwell, M., & Hughes, B. (2006). Frontiers in linguistic annotation for lower-density languages. In T. Baldwin, F. Bond, A. Meyers, & S. Nariyama (Eds.), *Proceedings of the workshop on frontiers in linguistically annotated corpora 2006* (pp. 29-37). Association for Computational Linguistics. https://aclanthology.org/W06-06

McEnery, T., & Hardie, A. (2011). *Corpus Linguistics*. Edinburgh University Press.

McEnery, T., & Hardie, A. (2013). The history of corpus linguistics. *The Oxford handbook of the history of linguistics*, *727*, 745.

Mitkov, R. (2001). Outstanding Issues in Anaphora Resolution. In A. Gelbukh (Ed.), *Computational Linguistics and Intelligent Text Processing* (pp. 110-125). Springer.

Mitkov, R. (2004). *The Oxford Handbook of Computational Linguistics*. OUP Oxford.

Mitkov, R. (2013). *Anaphora Resolution*. Routledge.

Moe, R. (2008). FieldWorks Language Explorer 1.0. *SIL Forum for Language Fieldwork 2008-011*. SIL Forum for Language. https://www.sil.org/resources/publications/entry/7793

Molina Mejía, J. M. (2021). *Lingüística computacional y de corpus: teorías, métodos y aplicaciones*. Editorial Universidad de Antioquia.

Moreno Sandoval, A. (1998). *Lingüística computacional: Introducción a los modelos simbólicos, estadísticos y biológicos*. Editorial Síntesis.

Nerbonne, J. (2007). Linguistic Challenges for Computationalists. In N. Nicolov, *Recent Advances in Natural Language Processing IV. Selected papers from RANLP 2005* (pp. 1-16). John Benjamins Publishing.

Padró, L., Collado, M., Reese, S., Lloberes, M., & Castellón, I. (2010). Freeling 2.1: Five years of open-source language processing tools. In N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odijk, S. Piperidis, M. Rosner, & Daniel Tapias (Eds.), *7th International Conference on Language Resources and Evaluation* (pp. 931-936). European Language Resources Association (ELRA).

Parodi, G. (2010). *Lingüística de corpus: De la teoría a la empiria*. Iberoamericana.

Pemberty Tamayo, J. L. (2020). *Concepción y elaboración de un sistema de etiquetado semiautomático para under-resourced languages* [trabajo de grado, Universidad de Antioquia]. Grupo de Estudios Sociolingüísticos]. Repositorio Institucional Universidad de Antioquia. https://bibliotecadigital.udea.edu.co/handle/10495/16570

Pemberty Tamayo, J. L. & Molina Mejía, J. M. (2020). UnderRL Tagger: Concepción y elaboración de un sistema de etiquetado semiautomático para Under-Resourced Languages. In J. M. Molina Mejía, P. Valdivia Martin & R. A. Venegas Velásquez (Eds.), *Actas III Congreso Internacional de Lingüística Computacional y de Corpus - CILCC 2020 y V Workshop en Procesamiento Automatizado de Textos y Corpus - WoPATeC 2020* (pp. 78-81). Universidad de Antioquia.

Pemberty Tamayo, J. L.; Molina Mejía, J. M. & Marín Morales, M. I. (2020). *UnderRL Tagger* (Versión 1.0) [Software]. Corpus Ex Machina, Universidad de Antioquia.

Pemberty Tamayo, J. L.; Molina Mejía, J. M. & Vallejo Zapata, V. J. (2023). UnderRL Tagger: un etiquetador gramatical para lenguas infrasoportadas tecnológicamente y lenguas minoritarias. *Forma y Función*, *36*(2). https://doi.org/10.15446/fyf.v36n2.101984

Poesio, M., Stuckardt, R., & Versley, Y. (2016). *Anaphora Resolution*. Springer.

Rogers, C. (2010). Review of fieldworks language explorer (flex) 3.0. *Language Documentation & Conservation*, *4*, 78-84.

Sáiz Noeda, M. (2002). Influencia y aplicación de papeles sintácticos e información semántica en la resolución de la anáfora pronominal en español. *Procesamiento del lenguaje natural*, *28*, 113-114.

Scannell, K. P. (2007). The Crúbadán Project: Corpus building for under-resourced languages. *Building and Exploring Web Corpora: Proceedings of the 3rd Web as Corpus Workshop*, *4*, 5-15.

Schmid, H. (1994). *TreeTagger-a language independent part-of-speech tagger*. https://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/

Schuster, S. & Manning, C. D. (2016). Enhanced English Universal Dependencies: An Improved Representation for Natural Language Understanding Tasks. In *LREC 2016*.

Straka, M. & Straková, J. (2017). Tokenizing, POS Tagging, Lemmatizing and Parsing UD 2.0 with UDPipe. In J. Hajič, D. Zeman (Eds.), *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies* (pp. 88-99). Association for Computational Linguistics. https://aclanthology.org/K17-3

Tognini-Bonelli, E. (2001). *Corpus Linguistics at Work*. John Benjamins Publishing.

Tordera Yllescas, J. C. (2011). *Lingüística computacional: Tecnologías del habla*. Publicacions de la Universitat de València.

Wallis, S. (2020). *Statistics in Corpus Linguistics Research: A New Approach*. Routledge.

Wilks, Y. (2010). Corpus Linguistics and Computational Linguistics. *International Journal of Corpus Linguistics*, *15*(3), 408-411.

Zeroual, I. & Lakhouaja, A. (2018). Data Science in Light of Natural Language Processing: An Overview. In J. Boumhidi, P. Érdi, Y. Ghanou, E. H. Nfaoui, & Y. Oubenaalla (Eds.), *Procedia Computer Science 127* (pp. 82-91). https://doi.org/10.1016/j.procs.2018.01.101