DeepString - Syntax Deep Explorer
Integrating multi-corpora support into a corpus analysis tool

João Trindade
joao.pedro.trindade@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

May 2020

Abstract

With the evolution of technology, linguists now have numerous tools to aid in their studies, including, *corpora* analysis tools. These tools help in determining how certain words are used in the context of a *corpus*, by calculating several association measures between co-occurrent words and by displaying these results in the form of a distributional profile.

One such tool is the Syntax Deep Explorer. This tool receives as input a *corpus* that was previously processed by STRING and allows the user to execute several searches with the syntactic information annotated on the *corpus*. STRING is a Natural Language Processing Chain for the Portuguese language developed by the Human Languages Laboratory on INESC-ID, that performs all the basic text processing tasks in natural language, including, syntactic analysis and the extraction of syntactic dependencies between constituents. Syntax Deep Explorer distinguishes itself from other *corpora* analysis tools by allowing searches based on these syntactic dependencies (subject, direct object, etc.) and by offering a more diversified array of association measures compared to other current tools.

This paper covers some improvements made and some new features implemented to Syntax Deep Explorer. The main implemented features that were developed are: the comparison between the distributional profiles of 2 words in the same *corpus* and the comparison of the distributional profiles of the same word in 2 distinct *corpora*; the presentation of examples, with the highlighting of target words, as well as, the improvement of the format in which distributional profiles are presented; and *multi-corpora* support. Two new *corpora* were constituted to support these new functionalities: a *corpus* from sports newspapers texts (*Desportivo*) and another with the minutes from the Portuguese Parliament (*Parlamento*).

Keywords: Natural Language Processing, Corpus Linguistics, Co-occurrence, Association measures, STRING

1. Introduction

A *corpus* analysis tool is usually built upon examining and extracting information from words in a *corpus*, relate them in some relevant way and present the user with some insightful knowledge about their relation. Using *corpus* analysis tools allows one to produce more apt examples and empirically better motivated descriptions of language use and strutcture in the sense that they use actual information from *corpora* to establish how words are used and discover patterns of a language, in a given *corpus*. The *Syntax Deep Explorer* (henceforward, for brevity, it will be referred to as *Explorer*) is one such tool that was developed in order to better analyze *corpora* processed by the Statistical and Rule-Based Natural Language Processing Chain (STRING)[10]. The Explorer differs from other *corpus* analysis tools by having multiple association measures for the user to choose from; and by applying them to syntactic dependency relations between words. This introduction is divided into three subsections: Subsection 1.1 describes the objectives of this dissertation; Subsection 1.2 gives a brief explanation of STRING; and Subsection 1.3 describes the original version of the Explorer.

1.1. Goal

The main goal of this dissertation is the improvement of the Explorer system, providing the user more information about the lemmas he or she is searching for, and increasing the number of available relevant features. To accomplish these goals this project focused on:

- Adding support for multiple *corpora*, simplifying the task of adding information to the Explorer database, and allowing the user to search for lemmas in different *corpora*;

- Adding the option to compare two different lemmas in the same *corpus*;
- Adding the option to compare the same lemma in two different *corpora*;
- Adding more information to each search;
- Improving the user experience by correcting visual inconsistencies and helping the user in finding relevant information.

1.2. STRING

STRING[1] is a Natural Language Processing Chain for Portuguese, developed at Instituto de Engenharia de Sistemas e Computadores - Investigação e Desenvolvimento (INESC-ID) Lisboa [10]. It has a modular structure, comprised of four main components.

- **LexMan: Lex**ical **M**orphological **an**alyzer
  LexMan [13] receives a plain text from the *corpus* and performs 3 tasks:
  1. **Tokenization.** LexMan divides the text into tokens. These are words, multiword units, numbers, punctuation, symbols and other textual units.
  2. **Token annotation.** Token annotation is the main function of LexMan. LexMan gives the token an annotation.
  3. **Sentence splitting.** This module splits the text into sentences. This is done by correctly identifying the sentences' boundaries.
- **RuDriCo2: Ru**le **Dri**ven **Co**nverter
  RuDriCo2 [5] is a rule-based morphological disambiguator that can also change the segmentation of the input. For example, it may receive the contracted form *no* 'in_the' and separate it into its constitutive elements, the preposition *em* 'in' and the define article *o* 'the' . Besides, RuDriCo2 also uses rule-based knowledge to morphologically disambiguate certain tokens. For instance, contextual rules are used to disambiguate the verbal participle *partido* 'broken' and the noun *partido*'political party'.
- **MARv: M**orphossyntactic **A**mbiguity **R**esol**v**er
  MARv [12] is a statistical Part of Speech (POS) disambiguator based on Markovian models, which uses the Viterbi algorithm to choose the most likely POS tag for each word. The language model used is based in trigrams, which encode contextual information; and unigrams, which encode lexical information.

  The probability of each tag is calculated based on a previously tagged, training *corpus* with approximately 250,000 words. This module offers a precision of over 97% when disambiguating words [10].
- **XIP: X**EROX **I**ncremental **P**arsing
  This module analyzes the morphologically tagged and disambiguated text as input and syntactically analyzes it using *Xerox Incremental Parser (XIP)* [1]. XIP parses the text dividing it into elementary phrases, known as *chunks* such as noun phrase (`NP`) or prepositional phrase (`PP`); and extracts syntactic relations (dependencies) between these chunks such as subject (`SUBJ`) or direct complement (`CDIR`). These dependencies are derived from a set of pre-programmed, manually crafted, syntactic rules, which constitute the Portuguese grammar of XIP. The rule-based grammar is divided into modules, which are ordered by their depth level. Rules with a lower depth level are applied first. With this method, it is possible to build highly detailed and rich lexical and dependency-based descriptions.

1.3. Syntax Deep Explorer

Syntax Deep Explorer[2] is a tool initially developed by José Pereira [11] at INESC-ID Lisboa in 2015. Its objective is to provide easy access for an analysis of co-occurrence patterns between words in order to understand how they are used in sentences taken from a given *corpus*.

As it was previously mentioned, the Explorer collects the output of STRING. This is done by saving the dependencies originated from the XIP module into a SQLite database and calculating several association measures for each word co-occurrence.

A previously developed XIP API [4] is used to access the output of XIP. This API transforms the XML output of XIP into the following Java structures:

- **XIPNode** is the basic structure of the chunk tree. A XIPNode can be a leaf or a branch on the chunk tree. If it is a branch, it contains children node, also represented as XIPNodes. A XIPNode also can contain a group of Features and Dependencies;
- **Token** represents the leaf XIPNode of a chunk tree. It presents information regarding the token that has been analysed by STRING, such as the word and the lemma;

[1] https://string.hlt.inesc-id.pt (last visited in 10 of April 2020)

[2] https://string.l2f.inesc-id.pt/demo/deepExplorer (last accessed in 27 of April 2020)

- **Feature** contains the properties of the XIPNodes or the properties of a Dependency;
- **Dependency** is a structure that contains information about the word co-occurrence produced in XIP;
- **XIPDocument** is the structural representation of a chunk tree, containing both XIPNodes and their respective Dependencies.

These dependencies form the core of this project: what is shown to the user are the most prominent word co-occurrences in certain dependencies between them, and according to the selected association measure. Each dependency relates two arguments, the *modified* or *governed* element, and the *modifying* or *governor* element. The XIP dependencies [2] used by the Explorer are:

- `SUBJ:` Associating a subject to a verb
- `CDIR:` Associating a direct (accusative) complement to a verb;
- `CINDIR:` Associating an indirect (dative) complement to a verb;
- `MOD:` Associating a word or expression to its modifier;
- `COMPL:` Associating a predicative element (e.g. a verb) to its essential complement;
- `QUANTD:` Associating a noun to its quantifier;
- `CLASSD:` Associating a noun to its nominal classifier.

To determine in which order the elements appear in the *corpus*, a property may be added to the end of each `MOD` dependency. If that property is `PRE` then the second element of the dependency appears on the text *before* the first element. For example, `MOD_PRE(homem,grande)` indicates that the word *grande* 'big' appears on the *corpus* before the word *homem* 'man' , as in the sentence: *Ele é um grande homem* 'He is a great man.' , whose meaning would be different if the adjective appeared after *after* the noun *Ele é um homem grande.* 'He is a big man.' .

`POST` indicates that the second word of the dependency appears *after* the first one, as it would be the case in the last example.

In addition to the `PRE/POST` properties added by STRING, this system also adds the POS of both words in the dependency. This addition is helpful when querying the database for specific dependencies. For example, `MOD_PRE_ADJ_NOUN` represents a dependency-property pattern where an adjective modifies a noun and the adjective appears before the noun in the sentence. In the case where the order of the co-occurrent words is unknown, the system adds a property called `SEM_PROP`, for example `SUBJ_SEM_PROP`.

In the graphical interface of the Explorer, the user inputs the lemma of the word to be searched; the POS of that word; the association measure to be used; the minimum number of occurrences required for a co-occurrence to appear in the results; and the maximum number of results that will appear on each section on the distributional profile (or *lexgram*, as it is called). At the moment, the Explorer only supports the search for nouns, verbs, adjectives, and adverbs. As it was previously mentioned, the Explorer differs itself from the other systems by its selection of association measure. The available association measures are Dice, LogDice, Pointwise Mutual Information (PMI), ChiPearson, LogLikelihood, Significance, and Frequency. The full description and rationale behind the choice of each measure is explained in [11], along with source references.

After selecting these options, and clicking the **Search** button, the system displays the co-occurrences of the chosen lemma separated into two columns, "on the left" and "on the right". Co-occurrence results are ordered by the chosen metric. Co-occurrences with a higher score in the chosen metric will appear first than co-occurrences with a lower score.

Next to each co-occurrence result, two values are provided, separated by a colon. The first value represents the binary logarithm of the frequency of the co-occurrence, while the second value is the score of the co-occurrence selected association measure.

The displayed information differs according to the chosen word class. For example, in Figure 1, it is possible to analyze the adjectives that modify the lemma *homem* 'man' and the words of which it depends as a complement. On the other hand, if the chosen lemma was an adjective, the Explorer would have displayed the adverbs that modify the chosen lemma and which nouns that adjective modifies.

After reaching the screen shown on Figure 1, the user can then click on any co-occurrence to get an in-depth look at the specific relation between these two lemmas. Clicking on that lemma will reveal a screen that shows some snippets of text from which the system extracted this specific relation.

Concluding this introduction, this dissertation project aims to improve the Explorer system by providing additional information, in the form of more dependencies and new features; and to better the user experience by reformulating certain ambiguous dependency titles.

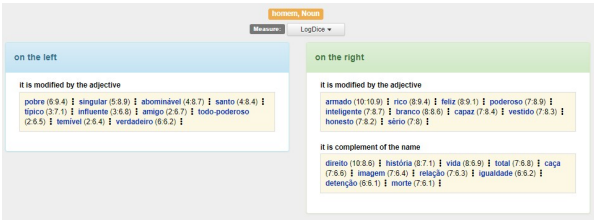

Figure 1: Co-occurrence profile (lexgram) of the noun *homem* 'man' using LogDice, with default values (from [11]).

2. Related Work

This section illustrates three different *corpus* analysis tools. Each one has its own way of displaying relevant data to the user and consequently, each may have a different utility. While the focus of most of the available systems is on providing examples of the use of a word in a given *corpus*, this project is built on a higher degree of abstraction. The analyzed systems are *DeepDict* [3],*CQPWeb* [7] and *Sketch Engine* [9].

2.1. Sketch Engine

Sketch Engine[3] is a commercially developed *corpus* analysis tool, created in 2003 by Lexical Computing, Ltd. This service provides most of the functionalities that have been also implemented in the Explorer, and overall is a more supported and more developed tool. This is why the main inspiration for the development of the Explorer has been drawn from Sketch Engine and its functionalities. The most relevant features that this service brings to the table are:

- access to various *corpora* and user inputted *corpora*;
- display of the lexical profiles (or *word sketches*) of a selected lemma. A lexical profile is a one-page, automatic, *corpus-derived* summary of a word's grammatical and collocational behaviour [9].
- comparison between the lexical profiles of two different words;

Sketch Engine's main feature is the *word sketch*, a comprehensive and detailed examination of how a chosen lemma is most commonly used in a certain language by showing the most frequently co-occurring words related to that lemma and how they fit together in a sentence.

Supporting the *word sketch*, the system provides the *concordance* for every use of the lemma. The *concordance* displays the context in which the lemma is used, with examples extracted directly from the *corpus*.

These results are aligned in KWIC format (keyword in context), the target lemma is placed at the center and the co-occurrent word appears on the left or on the right context. Both these words are highlighted.

The *word sketch difference* displays a hybrid *word sketch* between two different lemmas. The *word sketch* on Figure 2 shows the difference between *homem* 'man' and *mulher* 'woman' through a color-coded gradient, where darker green lemmas are more prominent when associated with the lemma *homem* 'man' and darker red lemmas are more often related to the lemma *mulher* 'woman' . According to the results on Figure 2 *homem* 'man' appears linked with words such as *moderno* 'modern' and *rico* 'rich' as opposed to *mulher* 'woman' , which is linked with such words as *lindo* 'beautiful' and *jovem* 'young' . The *word sketch difference* feature computes these results using only the frequency of the co-occurrences.

Figure 2: Part of the *Word Sketch Difference* between the lemma *homem* 'man' and the lemma *mulher* 'woman' .

The *thesaurus* functionality is another interesting feature in Sketch Engine's portfolio. It creates a list of words based on common collocations [9], that is, cases where both words are used in the same context, they would appear in each other's thesaurus. For instance, the word *livro* 'book' would appear in *jornal* 'newspaper' thesaurus and vice versa because they are both usually direct objects of the verb *ler* 'read' .

The last feature this tool covers is the option for the users to input their own *corpus*. When adding *corpora* to Sketch Engine, the users have two choices; uploading their own *corpus*, or finding texts on the web to constitute a new *corpus*.

2.2. DeepDict

DeepDict[4] is a graphical *corpus-based* dictionary of word relations developed at GrammarSoft and com-

[3]https://www.sketchengine.eu/ (last visited in 27 of December 2018).

[4]https://gramtrans.com/deepdict/ (last visited on April 21, 2020).

mercially released in September of 2007 [3]. This dictionary tool is studied below on the grounds that it was the main inspiration for the creation of Syntax Deep Explorer and this could offer additional insight on how to improve it.

A search made on this system results in a semi-graphical representation of a dictionary entry called a DeepDict *lexicogram* [3]. DeepDict also associates different templates to each word class, for grammatical reasons and to avoid ambiguities. Syntax Deep Explorer took inspiration from the layout of DeepDict's *lexicogram*: The relations of the target word with words that come before the chosen lemma will appear on the left of the screen and the relations that come after will appear on the right. In DeepDict, the order of the results is determined by a single association measure, a modified PMI score. Words with a higher PMI score appear first than words with a lower score. The Explorer also displays its results ranked by the selected association measure score.

When comparing the results given by the two systems it is possible to see some room for improvement in both of them. For example, Syntax Deep Explorer's noun template only shows to which words (in bold, below) the target lemma acts as a noun complement (e.g ***a vida*** *de um homem* 'the life of a man'); while DeepDict's noun template only shows the words (in bold, in the example) that act as complement of the target lemma (e.g *homem do* ***ano*** 'man of the year'). This difference is exemplified on Figure 1 and again on Figure 3.

Figure 3: DeepDict *lexicogram* of the word *homem* 'man' .

2.3. CQPweb

CQPweb[5] is a freeware *corpus* analysis tool, developed at Lancaster University, and it is made available via Lancaster University's CQPweb server. This tool resembles most of the tools available online and is one of the most popular and complete choices.

The main appeal of CQPweb is its flexibility and its support for POS tags [7]. There are two main query methods present in CQPweb-based systems, "Simple query" and "CQP syntax". CQP syntax, uses the Corpus Query Processor [6], a specialized search engine for linguistic research. CQP is a powerful and complex language but it can be daunting for the end user. The Common Elementary Query Language (CEQL) [8] was developed as a simpler alternative to CQP. It gives access to CQP's most used features but in a more user-friendly and accessible way. CEQL is used in CQPweb's Simple query.

The specialized corpus query languages and their flexibility allow for many sophisticated queries, beyond the scope of the other systems here mentioned. However, the time required, as well as the necessary knowledge and mastery of a relatively obscure query language to do so, makes this a less practical option. Having said that, the utility and versatility that CQP brings to the table is certainly useful when addressing the objectives of the Explorer.

3. Improving Syntax Deep Explorer

This section aims to describe the architecture of the implemented solution as well as the problems faced and the reasoning behind the chosen solution. The main goal for the improvement of the Explorer tool was to allow the user to compare the distribution profiles of two different words or of the same word in two different *corpora*, and to display the results in a way that is both legible and insightful.

3.1. *Corpora*

Selecting which *corpora* to use is important because the application ultimately relies on *corpora* to provide interesting and relevant results to the end user. For this project, the selected *corpora* were:

- **CETEMPúblico**[6], a *corpus* composed by extracts of texts from the Portuguese daily national newspaper *Público* from 1991 to 1998. This *corpus* is available to the public and contains 175,350,145 words;
- **Parlamento**, a *corpus* composed by minutes from sessions of the Portuguese Parliament from 1976 to 2018. This *corpus* contains 123,633,859 words;
- **Desportivo**, a *corpus* specifically created for this project. It is composed by texts from the Portuguese sports newspapers *O Jogo*, from 1999 to 2005; and *A Bola*, from 2000 to 2006. This *corpus* contains 100,161,374 words.

These *corpora* were selected mainly due to their size and the fact that their source texts were already available at the INESC-ID. These *corpora* having different domains was also a benefit since really different writing styles are expected

[5]https://cqpweb.lancs.ac.uk/ (last visited in 21 of April 2020).

[6]https://www.linguateca.pt/cetempublico/whatisCETEMP.html

from these domains and, hopefully, it will provide in more diverse results in *corpora* comparisons. The Parlamento *corpus* had been previously processed by STRING, so it was ready to be used for this project. CETEMPúblico also had been previously processed by STRING. However, an updated version of STRING processing chain and of its Portuguese grammar (in XIP) were now available, and since the Explorer aims to be up-to-date with STRING developments, this *corpus* was reprocessed. The Desportivo *corpus* was created from texts from two different sports newspapers, *O Jogo* and *A Bola*. To prepare these sports newspapers texts for a STRING analysis, some processing was done, namely: converting both sets of texts to UTF-8 encoding; removing irrelevant information and advertisements; appending unfinished, splitted sentence fragments back together; inserting common unknown words into STRING's dictionaries; and correcting some typos on the *corpus*.

Table 1 is a comparison between the different *corpora* available on the Explorer. While the sizes of Desportivo and Parlamento are very similar, the CETEMPúblico *corpus* is almost twice as large as these two, in terms of size on the database and total number of different co-occurrences and words. In practice, this will result in slower searches on CETEMPúblico, when compared to the other two *corpora*.

Corpus	CETEMPúblico	Parlamento	Desportivo
Size on the database	6.28 Gb	3.47 Gb	3.64 Gb
Time to load on the database	4 days, 21 hours and 21 minutes	2 days, 22 hours and 6 minutes	2 days, 23 hours and 59 minutes
Files	4,042	5,175	3,312
Different co-occurrences	8,108,800	4,281,650	4,028,456
Different words	249,100	112,387	129,439
Different nouns	180,674	70,036	92,928
Different verbs	16,303	11,083	10,840
Different adjectives	46,837	26,671	22,207
Different adverbs	5,286	4,597	3,464

Table 1: Constitution of the chosen *corpora*.

3.2. Comparison

The comparison of distributional profiles is the main contribution of this dissertation. This subsection aims to describe in-depth how this comparison is made, and the criteria to make a result appear on the user's screen. The response time of the comparison of distributional profiles is evaluated later on Subsection 4.2.

Two types of comparisons can be executed in this solution: a comparison where the user selects two different lemmas and the Explorer compares them in the context of a single *corpus* (*Word Comparison*); and a comparison where the user selects just one lemma to be compared across two different *corpora* (*Corpora Comparison*). The logistics behind these two comparisons is very similar. A comparison consists of two different Explorer searches, the only difference being the limit of the SQL query. While a normal Explorer search has a limit set by the user selected via the *Maximum number of co-occurrences to display* option in the main screen, a comparison search does not utilize this value directly. Similarly, the *Minimum Frequency* established by the user in the main screen is not used directly in the query. The purpose of this change is for the search to return every result in order to achieve a more accurate comparison.

Both searches are returned in full to the Javascript part of the program were they are pruned to display only the relevant information to the user. The program iterates over the positions of the *first* search (`PRE_WORD, POST_WORD, PRE_VERB, POST_VERB`) and the type of dependency (`SUBJ_SEM_PROP`, etc.), and, in every iteration, it executes the following changes: for each lexical profile, select the X results with the highest selected association measure to appear on the screen, with X being the value the user selected in the *Maximum number of co-occurrences to display* option on the main menu; determine, for each word, the value of the association measures and the percentage score that will appear next to it; and eliminate results that do not meet the *Minimum Frequency* set by the user. A percentage score is calculated using the expression below:

$$\frac{\frac{fb_{cooc}}{fb_{totalcooc}}}{\frac{fb_{cooc}}{fb_{totalcooc}} + \frac{fr_{cooc}}{fr_{totalcooc}}} \times 100 \qquad (1)$$

where $\frac{fb_{cooc}}{fb_{totalcooc}}$ is the relative frequency of the co-occurrence between the first lemma of the comparison and the co-occurrent lemma and $\frac{fb_{cooc}}{fb_{totalcooc}}$ is the relative frequency of the co-occurrence between the second lemma of the comparison and the co-occurrent lemma. This expression aims to form a ratio between these two relative frequencies, in order to show the user which co-relation is quantitatively more important, while also taking into account the differences in the total number of occurrences of the two different lemmas, or the same lemma in two different *corpora*.

As a result of the methodology used to execute the comparison, the symmetry of the comparison can be assured, meaning that changing the order of the lemmas only results in aesthetic differences, namely

the color associated with each lemma; and, since the results are ordered by their normalized frequency, the order in which they appear also changes. The co-occurrences that appear on a simple Explorer search for a given lemma will always appear on a comparison of this lemma (assuming the chosen options are the same).

3.3. Web Interface

This subsection emphasizes the differences between the Explorer's old version and the new one.

The main screen was changed in order to allow the user to select a *corpus*, in which to search for the desired lemma and to allow the user to compare two lemmas in a *corpus* or to compare the same lemma in two different *corpora*. Two buttons were added to allow the user to compare different words and different *corpora*. Activating a comparison box displays an additional row to the user. This row contains only one additional field to be filled by the user while all the other redundant fields are greyed out since they must be the same as the ones on the first row, they only act as a reminder to the user. The field that is not greyed out depends on the desired comparison. If the user requests a *Word Comparison* by clicking on the button named *2 Words / 1 Corpus*, he or she has to provide the additional lemma to compare. If the user requests a *Corpora Comparison* by clicking on the button named *1 Word / 2 Corpora*, he or she has to provide the additional *corpus* to compare.

Some changes were made to the result screen in order to improve the readability of the user interface, for example, changing the names of dependencies, changing ambiguous terms on the interface and adding the number of co-occurrences of the target word. The interface was also reconfigured in order to better convey a sense of symmetry and cohesion between result screens. For example, since the verb result screen shows which nouns act as direct complement to that verb, the noun result screen should also show to which verbs that noun acts as direct complement. The impact that these additions have on the response time of the Explorer is evaluated on Subsection 4.2.

Figure 4 depicts the result screen of a *Word Comparison*, i.e. a comparison between two different lemmas in the same *corpus*. The main differences between this screen and the simple search screen are the header and the values next to each co-occurrence result. When comparing two different lemmas in a given *corpus*, the header of the result screen has five different components: the type of comparison, in this case being a *Word Comparison*; the name of the *corpus* where the comparison is taking place; a drop-down menu indicating the measure being used and allowing the user to change it; on the left side, enveloped by a blue rectangle is the first lemma the user inserted and, at its' right, the number of co-occurrences of this lemma in the chosen *corpus*, present in the database; and, on the right side, enveloped by a red rectangle is the second lemma the user inserted and, at its' right, the number of co-occurrences of this lemma in the chosen *corpus*, present in the database;

The result screen of a *Corpora Comparison* is also very similar. The main difference being the header of the result screen, which instead of displaying the compared lemmas on the colored rectangles, it displays the name of the *corpora* being compared.

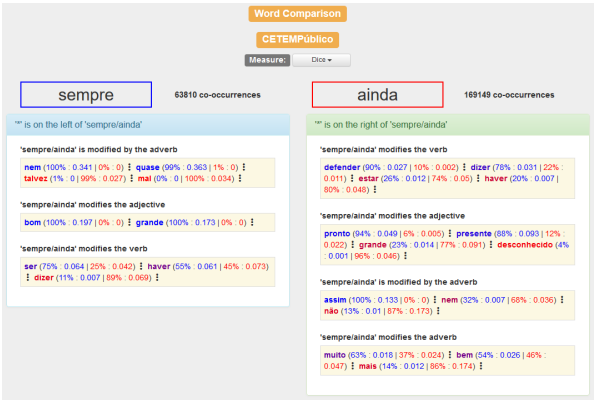

Figure 4: Word comparison between adverbs *sempre* 'always' and *ainda* 'yet' in the CETEMPúblico *corpus*.

Clicking on one of the co-occurrences on the comparison result screen generates a popup with a detailed view of that co-occurrence with additional information and example sentences like in the search result screen. Figure 5 illustrates an example of a detailed view of a co-occurrence result in a comparison between two words (adjective *pronto* 'ready' and adverbs *sempre* 'always' and *ainda* 'yet'). The choice of colors is used once again on the lines beneath the title to indicate which part of the screen belongs to each element of the comparison.

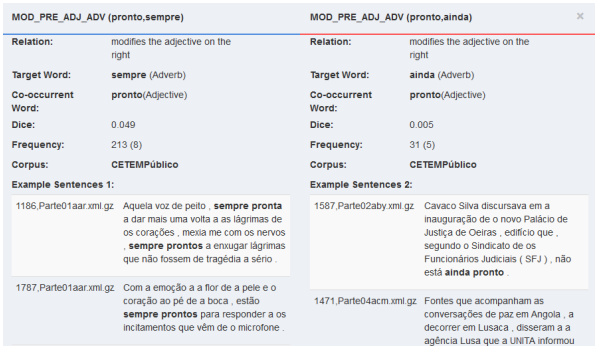

Figure 5: Detailed view of a co-occurrence comparison.

Finally, the Help[7] and About[8] pages of the Explorer were updated to reflect the changes made on the system.

3.4. Word highlighting in example sentences

Another feature that deserves attention is the word highlighting in example sentences. Highlighting the target and co-occurrent words allows for a faster inspection of the example sentences (Figure 5). The original version of the Explorer didn't display this feature. Highlighting a word on the original version of the Explorer is not a trivial task since there were various setbacks and several different ways to develop this feature. This subsection describes the approach taken to highlight the searched word and the co-occurrent word in the example sentences.

The biggest hurdle when highlighting these words is the fact that the system registers only the lemmas present in the co-occurrence while the example sentences may contain an inflected form of those lemmas. So, the challenge consisted in selecting the inflected form of a lemma from a string while only having the lemma to work with. The developed solution tries to match the inflected form to its lemma by using a regular expression. This regular expression removes the last letter from the lemma and is used to test every word in the example sentence for a match, if a match occurs that word is highlighted. So if we have the lemma *funcionário* 'employee' the corresponding regular expression will be **funcionári*** and, for example, will highlight words such as *funcionário*, *funcionária*, *funcionários* and *funcionárias*.

This solution has two major flaws. It is possible to highlight words other than the target word and co-occurrent word, and it is also possible not to highlight the co-occurrent word or the target word (or both). For example, when searching for the lemma *ser* 'to be' , the corresponding regular expression will be **se*** which will match correctly to words like *ser*, *seja* or *será*. However it will miss words such as *foi*, *era* or *é* which are all conjugations of the verb *ser* and match with words such as *serra* 'saw' , *semente* 'seed' or *sempre* 'always' .

Despite its issues, this was the algorithm chosen in this project to highlight co-occurrent words in example sentences. The accuracy of this algorithm and the impact it has on the usability of the system is evaluated in Section 4.1.

4. Results

In this section, the evaluation method of the newly developed features is described as well as the obtained results. Subsection 4.1 evaluates the performance of the word highlighting algorithm presented in Subsection 3.4, and Subsection 4.2 evaluates the performance of the web application, described in Subsection 3.3.

4.1. Word highlighting in example sentences

This functionality was subjected to two different tests. The first test measured the accuracy of the word highlighting algorithm and the percentage of times a word was incorrectly highlighted in the test group. The second test was a user test with the goal of measuring the impact of this functionality on the user experience.

The Explorer allows the search for 4 different POS, and so the test sample consists of the 10 nouns, verbs, adjectives and adverbs with the largest number of co-occurrences (ignoring named entities).

An Explorer search was made for every lemma in the test sample with a *Maximum number of co-occurrences to display* set to 5 and selecting the CETEMPúblico *corpus* with the Dice metric. For every co-occurrence resulting from the search, the first example sentence was extracted. Since each POS may enter in different dependency relations, the resulting number of extracted sentences was different for each one. This testing sample consisted of: 476 sentences where the target word is a noun; 358 for verbs; 220 for adjectives; and 310 sentences for adverbs. Table 2 show the result of this explanation.

Each sentence gained one accuracy point when either the target word or the co-occurrent word is highlighted; and two accuracy points were given when both words are highlighted; conversely, no accuracy points were given when the algorithm failed to highlight either word. The *Accuracy* row on Table 2 represents the amount of points each POS had divided by the maximum number of possible points (two times the number of sentences for each POS). The *Error rate* row represents the number of sentences where one or more words, other than the target word or the co-occurrent word, have been highlighted, divided by the total number of sentences for each POS.

POS	Nouns	Verbs	Adjectives	Adverbs
Accuracy	85%	48%	67%	87%
Error rate	11%	18%	11%	21%

Table 2: Evaluation of the highlighting algorithm.

When analysing the results, certain factors that lower the accuracy of the algorithm became evident, namely: the 1990 Portuguese spelling reform; n-grams with size 2 or more (because the algorithm

[7] https://string.hlt.inesc-id.pt/demo/deepExplorer/#/help (last visited in 22 of April 2020).

[8] https://string.hlt.inesc-id.pt/demo/deepExplorer/#/about (last visited in 22 of April 2020).

splits the text into single words); and verbs and adjectives with irregular forms.

Table 3 represents the average amount of time a user takes to find the target word and the co-occurrent word in example sentences. This test was done using the word highlighting algorithm and with no highlighting, (as a control). Since the word highlighting algorithm has a varying performance depending on the POS of the words, as it has been seen on the previous test, the words to test contain examples from the four different POS. In this test, a group of 10 users were recruited and each person would have to point to both the target and the co-occurrent word in 48 sentences divided into 2 sets of sentences named Group A and Group B. In Group A, none of the sentences were highlighted and in Group B all the sentences were highlighted following the word highlighting algorithm. Some sentences in Group B contained incorrect highlighting and missing highlighted words to better simulate the functioning of the highlighting algorithm. Table 3 presents the results from the user tests.

POS	Noun	Verb	Adjective	Adverb
Group A	5.19	5.25	4.44	6.79
Group B	2.49	2.46	2.73	2.88

Table 3: Average lookup time of both test groups (in seconds).

Despite the imperfect accuracy and the incorrect highlightings of the algorithm, the lookup time decreased, on average, 51%.

4.2. Web application

This section will contrast the response time between the original Explorer (V1) with the one modified in this project (V2) as well as measure the time needed to execute a comparison. Each search was executed on a personal laptop, with an Intel Core i5-8265U CPU, 8 Gb of RAM and running an Apache server locally on a SSD storage device. These tests were performed locally in order to mitigate the impact of internet issues in the results.

To evaluate the time needed to execute a search, two lemmas from each part of speech were randomly selected (one lemma with a significantly higher frequency than the second lemma). Each lemma was searched 3 times on the Explorer and the average time was inserted into Table 4. The response time was measured using the *Network Monitor* present in the *Mozilla Firefox* web browser version 75.0. The database used for both systems was the same and both systems were deployed locally.

As expected, the changes made to the system slightly increased the response time (in average 12,64 %). In every POS there is now an extra SQL query for the total number of co-occurrences

POS	Noun	Verb	Adjective	Adverb
ExplorerV1	53	238	38	69
ExplorerV2	82	243	41	74

Table 4: Average time (in ms) of each version of the Explorer by POS.

of the target lemma. In addition to this, the noun and verb result screens have additional dependency-property patterns, which results in more queries for those particular POS, thus increasing the response time even further, especially the noun.

The Word Comparison introduced in this dissertation was evaluated in a similar manner. For each POS, 3 sets of lemmas were selected with varying frequency of co-occurrence. Each comparison was executed 3 times and the average time is presented in Table 5. The response time was measured in the same way as in the previous evaluation and the *corpus* used was the CETEMPúblico.

POS	Noun	Verb	Adjective	Adverb
Time	357	919	332	204

Table 5: Average time (in ms) of a Word Comparison by POS.

Despite the obvious and expected increase in response time, the Explorer manages to deliver a word comparison in less than a second.

Table 6 compares the difference in response time in a Corpora Comparison. Each pair of *corpora* is tested with the same words to better understand the difference in response time under the same conditions. Once again, each POS was tested 3 times, with lemmas with high and low frequency, and the resulting average time is presented below.The response time was measured in the same way as in the previous evaluations.

The response time in the first two columns is very similar since the Desportivo and Parlamento *corpora* have similar sizes. The main difference comes between the first two columns and the third one. Since the CETEMPúblico *corpus* is twice as large as Desportivo and Parlamento, comparisons featuring CETEMPúblico are expected to take a longer time than comparisons that do not involve this corpus.

Corpora	CETEMPúblico and Desportivo	Parlamento and CETEMPúblico	Desportivo and Parlamento
Noun	529	604	393
Verb	2,106	2,171	1,458
Adjective	223	233	96
Adverb	411	441	359

Table 6: Time (in ms) to execute a *corpora* comparison for different words.

5. Conclusions

This dissertation covered some improvements introduced in the *corpora* analysis tool DeepString - Syntax Deep Explorer. The main added features were the support for multiple *corpora*; the comparison between words within the same *corpus*, and the comparison of the same word accross two different *corpora*; and the algorithm for highlighting target words in the example sentences.

To add support for multiple *corpora*, the database was changed in order to be more modular and facilitate the addition of new *corpora*, as well as changing or removing existing *corpora*. This also aims at improving the response time of the SQL queries, should the database grow to a much more considerable size.

Two new *corpora* (*Desportivo* and *Parlamento*) were added to the system, one of which had to be built almost from scratch before being processed by STRING.

The comparison feature took inspiration from Sketch Engine's comparison, but expanded upon this concept by allowing different association measures to be compared, and by allowing the user to compare a lemma in two different *corpora*.

The word highlighting algorithm, although still rudimentary, proved to be a helpful feature, drastically reducing the time that a user needs to read the example sentences, even when not providing the most accurate results. The changes applied to the interface also improved the consistency of the placement of dependencies and reduced the ambiguity found in certain cases.

References

[1] S. Aït-Mokhtar, J.-P. Chanod, and C. Roux. Robustness beyond shallowness: incremental deep parsing. *Natural Language Engineering*, 8(2-3):121–144, 2002.

[2] J. Baptista and N. Mamede. Nomenclature of chunks and dependencies in Portuguese XIP Grammar 4.6. Technical report, L2F-Spoken Language Laboratory, INESC-ID Lisboa, Lisboa, 2016.

[3] E. Bick. DeepDict–a graphical corpus-based dictionary of word relations. In *Proceedings of the 17th Nordic Conference of Computational Linguistics (NODALIDA 2009)*, pages 268–271, Odense, Denmark, May 2009. Northern European Association for Language Technology (NEALT).

[4] F. Carapinha. Extração automática de conteúdos documentais. Master's thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa, 2013.

[5] C. Diniz, N. Mamede, and J. D. Pereira. RuDriCo2-a faster disambiguator and segmentation modifier. *II Simpósio de Informática (INForum), Universidade do Minho*, pages 573–584, 2010.

[6] S. Evert and A. Hardie. Twenty-first century Corpus Workbench: Updating a query architecture for the new millennium. *Presentation at Corpus Linguistics 2011, University of Birmingham*, 2011.

[7] A. Hardie. CQPweb—combining power, flexibility and usability in a corpus analysis tool. *International Journal of Corpus Linguistics*, 17(3):380–409, 2012.

[8] S. Hoffmann, S. Evert, N. Smith, D. Lee, Y. Berglund-Prytz, et al. *Corpus linguistics with BNCweb-a practical guide*, volume 6. Peter Lang, Bern, Switzerland, 2008.

[9] A. Kilgarriff, V. Baisa, J. Bušta, M. Jakubíček, V. Kovář, J. Michelfeit, P. Rychlỳ, and V. Suchomel. The Sketch Engine: ten years on. *Lexicography*, 1(1):7–36, 2014.

[10] N. Mamede, J. Baptista, C. Diniz, and V. C. ao. STRING - A Hybrid Statistical and Rule-Based Natural Language Processing Chain for Portuguese. In *PROPOR 2012*, Coimbra, Portugal, 2012. PROPOR, PROPOR.

[11] N. Mamede, J. Correia, and J. Baptista. Syntax Deep Explorer. In *Computational Processing of the Portuguese Language: 12th International Conference, PROPOR 2016, Tomar, Portugal, July 13-15, 2016, Proceedings*, volume 9727, page 189. Springer, 2016.

[12] R. Ribeiro. Anotação morfossintáctica desambiguada do português. Master's thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa, 2003.

[13] A. Vicente. Lexman: um segmentador e analisador morfológico com transdutores. Master's thesis, Instituto Superior Técnico-Universidade Técnica de Lisboa, Portugal, 2013.