Syntax Deep Explorer

José Miguel Pinheiro Correia

Instituto Superior Técnico, Universidade de Lisboa
L^{2}F – Laboratório de Sistemas de Língua Falada – INESC ID Lisboa
Rua Alves Redol 9, 1000-029 Lisboa, Portugal

Abstract. The analysis of the co-occurrence patterns between words allows due to better understand the use (and meaning) of words its most straightforward application are lexicography and linguist description in general. There are already some tools that allow to obtain this co-occurrence information for any word in Portuguese corpora. This tool adopt different association measures to quantify each cooccurrence.
The presented solution consists in the extraction of cooccurrences statistics from corpus processed by STRING, and to provide a graphical interface to access the cooccurrence information stored in a database, using different association measures.
In this way, this project will allow the development of finer lexical resources and the STRING processing in general, as well as providing public access to co-occurrence information derived from these systems' parsing modules.

Keywords: Natural Language Processing (NLP), Graphic interface, Co-occurrence, Colocation, Association measures, STRING

1 Introduction

The analysis of the co-occurrence patterns between words in texts, allows to understand the differences in use (and meaning) that are associated with the different grammatical relations in which a word can participate. The quantification of these patterns is a powerful tool in modern lexicography as well as in the construction of basic linguistic resources for the processing of natural language, like thesauri.

The stakeholder to the study these patterns are linguists, languages students, translators and corpora linguistic researchers who study the behaviour of linguistic expressions. For all of these, co-occurrence data are essential for a better understanding of the language.

For better analysis of co-occurrence patterns, it is important to provide an interface that helps the user to explore the patterns extracted from a corpus.

The problem identified is the absence of a tool that collect the cooccurrence patterns derived from based syntactically analysed texts, and specifically from a corpus processed by STRING (STatistical and Rule-based Natural lanGuage processing chain) [14].

STRING is a hybrid statistical and rule-based natural language processing for (NLP) chain for the Portuguese language, developed by INESC ID's Spoken Language Systems Lab (L^2F). This has a modular structure and performs all basic text processing tasks. The first module is the LexMan [25], which is responsible for assign to each token its part-of-speech and any other relevant morphosyntactic features. The module RuDriCo [10] is a rule-driven converter, used to revolve ambiguities and to deal with contractions and certain compounds words. The module MARv [19], the statistical based on HMM and part-of-speech disambiguator, that chooses the most likely POS tag for each word, using the Viterbi algorithm. The XIP (Xerox Incremental Parser) [1] uses a Portuguese rule-base grammar to divide the text into chunks, and to establish syntactic dependency relationships between them. The result obtained in STRING goes through an anaphora resolution and temporal expressions normalization [14]. The processing results in a XML file.

The XIP syntactic dependencies analysed in this project are [15]: `SUBJ` dependency, which links a verb and its subject; `CDIR` dependency, linking a verb and its direct complement; `CINDIR` dependency, linking the verb with a prepositional phrase; `MOD` dependency that links a modifier with the modified element[1]; `COMPL` dependency that links a predicate (verb, noun or adjective) to each of its essential complements; `QUANTD` dependency that links a nominal head and a quantifier and `CLASSD` dependency that links a nominal head and a nominal classifier. A semantic dependency, Named Entity (`NE`), was also considered. This dependency delimits named entities and assigns them to several general categories.

1.1 Goals of the Project

The aim of the project is to develop a tool that, based on the output of the natural language processing chain STRING, allows to obtain easy and efficient access to co-occurrence data obtained from Portuguese texts and, for each co-occurrence, quantify it with several association measures. The information on these patterns is made available through a web interface, that organizes the information for a simpler analysis by users.

2 Related Work

2.1 Association Measures

In this project, it is essential to quantify the co-occurrence patterns. Six different association measures commonly used in corpus-based linguistic and cooccurrence analysis, were studied. The first association measure is Pointwise Mutual Information (PMI) [7], which links the number of co-occurrences between two words with the number of occurrences of each word.

The Dice Coefficient [9,24] is another association measure that calculates the degree of cohesion between two words . The measure LogDice [21] is an variant of the Dice measure and fixes the problem of very small calculated values.

[1] This is an umbrella dependency, as it also corrects PP complements to their governor.

The Pearson's chi-squared measure [16] is a statistical method of hypothesis testing against the null hypotheses. This association measure compares the observed frequencies with the expected frequencies for a given pattern in order to verify the independency between events. If the difference between observed and expected frequencies is large, then the null hypotheses is rejected. This measure shouldn't be used in small corpora, however.

The Log-likelihood Ratio [11] is another approach to hypothesis testing, and it is more suitable for sparse data than the previous method. Two hypotheses are defined in the beginning and the result shows which of them is more likely to happen. The hypothesis 1 says that the occurrence of a word is independent of the occurrence of another null hypothesis. The hypotheses 2 says that the two words occur together in a significant way.

The last one, the Significance Measure [4], is comparable to the statistical method *G-Test* for Poisson distribution. This measure associates the co-occurrences of two words with the number of sentences where they occurred.

2.2 Systems

Nowadays, there are already some few tools that allow due to get the information on the co-occurrence patterns of a word in Portuguese corpora, like DeepDict [3], SketchEngine [12] and Wortschatz [4].

The DeepDict is a tool developed by GrammarSoft[2] and presented in Gramtrans[3] platform. For Portuguese corpora, this tool uses the NLP analyzer PALAVRAS [2] and for the dependencies analyzer, `dep-grams` are collected. A `dep-gram` is a pair of lemmas of two words that represent a relation between them. The *dep-grams* are quantified by PMI measure [3]. In the database, the tool stores the `dep-gram`, the value of PMI, absolute frequency and the ID of corpus sentence where these occurred [3]. The DeepDict has a simple form as an interface, where the user can search any lemma. The result is presented as a lexicogram of the search lemma, that shows the different dependency relations that the word estabishes with other lexical elements, sorted in descending order by value of PMI measure. Each lexicogram admits the natural lecture of the words' relation. Words having different part of speech (POS) have different lexicograms [3].

The Sketch Engine is a tool developed by Lexical Computing[4]. This system uses Manatee [20] to manage the corpora. The tool stores for each word its lemma and POS tag [13]. Co-occurrences are identified by the tool and are quantified by the LogDice measure [13]. The system has a graphical interface that is generated by the tool Bonito [20] and allows access to stored data. The system's main features are: (i) *Concordance*, access to examples that show the corpora data without any analysis; (ii) *Word Sketches*, which show the word's grammatical behaviour and how it relates to others; (iii) *Sketch-diff* that shows word sketch

[2] `http://grammarsoft.com/` (last visit 11/10/2015).
[3] `http://gramtrans.com/gramtrans` (last visit a 11/10/2015).
[4] `http://www.sketchengine.co.uk/?page=Website/Company` (last visit 11/10/2015).

differences between two different words and (iv) *thesaurus* that shows similar words for a search word [12]. The system allows the users to create and manage their own corpora.

The project *Wortschatz* is a system develop by Leipzig University, which creates lexical similarities' networks from corpora. The system identifies two types of co-occurrences: where words occurring together in a sentence and where words occurring next to each other [18]. The system has efficient tree-based algorithms that allows a co-occurrence analysis of the complete corpus in quick time [4]. The system does not use the lemma of a word to join words together. Each co-occurrence is quantified by a Significance Measure and it is stored in a MySQL database. The system has an interface that allows the access to a word; its frequency class; examples of its use (sentences); significant co-occurrences and co-occurrence chart [18].

To complete this section, we can observe that all systems are able to list the co-occurrences of a word. The Sketch Engine is the system that offers more features, however the DeepDict shows the co-occurrences between words' information in a more organized way.

3 Solution Architecture

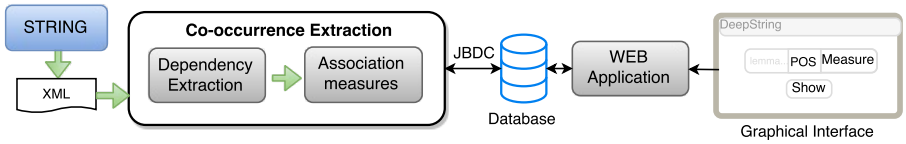

Fig. 1. Solution Architecture.

Figure 1 summarizes the solution's architecture, which consists in three modules: the database; co-occurrence extraction module and the web application.

3.1 Database

In this project, to store the co-occurrences extracted from corpora, a consistent data model with low information's redundancy is required. Thus, Entity-Relationship (E-R) Model [6] was developed, because this model allows a more natural perception of the problem. A `entity` has a set of attributes, a `set of entities` group those that have the same attributes. These can relate to each other. Each `relationship` has attributes as well [6].

The Figure 2 represents the E-R model for the system's database. The data integrity limitations for this model are:

DI1: Co-occurrence's words must belong to the same `Corpus` the `Co-occurrence` relationship is related to;

DI2: The `Co-occurrence` relationship must be related to the same `Property` which the words are related in the `Belongs` relationship;

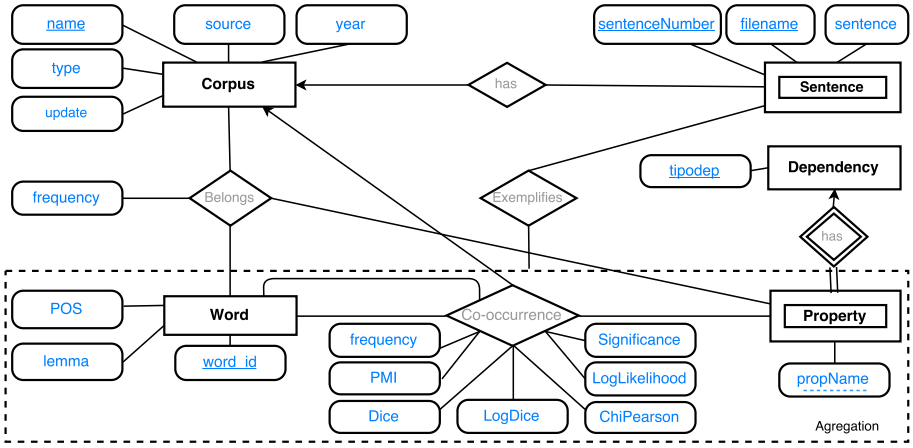

Fig. 2. Entity-Relationship Model.

DI3: The sentences in the `Exemplifies` relationship must belong to the `Corpus` which is related to the `Co-occurrence`.

Since typical database systems use a relacional model [8], it is necessary to convert the E-R model. The relacional model is defined by a set of relations, represented in tables, where the columns are attributes of the relationship and each row is a tuple (entry), which is unique[8]. To identify each tuple, primary keys are used to distinguish them. If a relation references another, it has reference attributes called *foreign key*, where the values of these attributes have the *primary key* value of a tuple in the referenced relation [23].

The conversion to the relational model must follow the defined conversion rules and it must follow the previous data integrity constraints.

The SQLite[5] was used to implement the obtained relational model. SQLite works locally, and allows a direct and faster access to data, without a remote server. The access to the data is made through SQL language, which provides the capability of manipulating the tables present in the database in order to get the results.

3.2 Co-occurrence Extraction

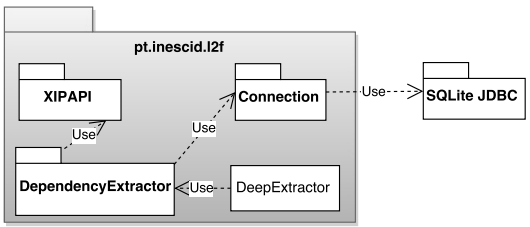

Fig. 3. *Packages* diagram for dependency extraction.

[5] `http://www.sqlite.org/about` (last visit 28/8/2015).

Figure 3 introduces the UML packages diagram for a tool which is responsible for the extraction of co-occurrences, the `DeepExtractor`. The diagram shows the main components used.

The XIPAPI [5,17] is an API that enables transformation of the XML files content from the XIP module of STRING in Java structures. The XIPAPI organizes the information imported. The result is the `XIPDocument` that consists in a set of `XIPNode`s and `XIPDependency`. A `XIPNode` represents a node from XIP, the basic element of `chunks`' tree. It contains another `XIPNode`s (child nodes) and `Features` with the properties of the node. A `XIPDependency` contains the information about a detected dependency in XIP and its `XIPNode`s which applies. [17].

To store the data in database it's necessary to connect it. For this, it's used an API called JBDC (Java Database Connectivity) that provides a Java program to a SQL database. In this project, the SQLiteJDBC[6], the JDBC for SQLite, is used.

The package `Connection` was developed with the purpose of facilitating the access to project's database. In the package a connection to the database through the SQLiteJDBC is established. Here, a group of classes was also been developed where the methods are used to manipulate the database's information. Each class represents a database's table and has the methods to verify if a particular entry exists and inserts new entries to that table. In case of classes responsible for `Belongs` and `Co-occurrence` tables, they have additional methods that will allow the calculate of the values from the association measures.

The package `DependencyExtractor` provides the core of the extraction tool. With the help of XIPAPI, the package analyses the texts produced by STRING to collect the co-occurrences. The `DependencyExtractor`'s method `Extract`, that accepts a `XIPDocument`, for each sentence (`XIPNode TOP`) in this document obtains in the sentence: set of named entities; set of `XIPDependency`, each one with two `XIPnode` and the string that is represented the sentence.

Afterwards, for each `XIPDependency`, the analysis process depends on the type of dependency it belongs to. A set of classes was developed, each one represents a XIP dependency (`MOD`, `SUBJ`, `CDIR`, `CINDIR`, `COMPL`, `QUANTD` and `CLASSD`) and it runs the correspondent code. All these classes serve an abstract class, the `DependencyType`, which implements the methods in common.

In these classes, the method `getDepInformation` accepts a `XIPDependency` and a set of named entities in the sentence. In this method, to prevent the analysis of dependencies and properties that are not relevant to this system, the dependency-property pattern of `XIPdependency` is checked in a list of predefined patterns. If that pattern is present, the analysis proceeds normally. The list of dependency-property patterns present in the system indicates for each word's POS which are the relations stored in the database. Then, for each `XIPNode` the word's lemma and POS are obtained. In the case that a `XIPNode` is present in the set of named entities, the word's lemmas are replaced by the correspondent named entity's category.

[6] `https://bitbucket.org/xerial/sqlite-jdbc` (last visit 07/09/2015).

During the process, it is necessary to store the information that is being found. Once a processed corpus is divided into several files, for each one, the information extracted is gathered in Java structures. The `DeepStorage` class works as a local database inside the execution and helps to organize the information until it is stored in the database. After the corpus' file extraction is completed, all gathered information is then stored in the database.

When the corpus extraction is completed, it is necessary to calculate the cohesion's value between the two words in a stored co-occurrence. The `DeepMeasures` class starts the calculation process. The package `Connection` is used to access the co-occurrences information in the database, and then, calculate and store each one of the six measures described above.

3.3 Web Application

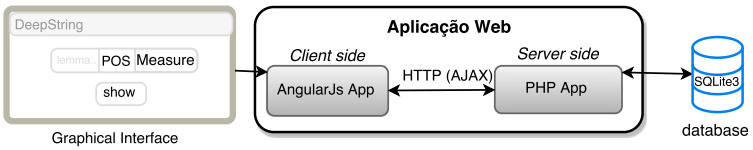

Fig. 4. Web application architecture.

Figure 4 shows the architecture of the web application. It has two main components, the server-side and the client-side (browser). The first runs the code that implements the process of information in the database and the second runs the code that implements the graphical interface. The communication between them is performed through asynchronous AJAX (Asynchronous Javascript and XML) requests (via HTTP) and the transmitted data is a JSON (JavaScript Object Notation) object.

In the server-side, the code was developed in PHP. From a client's request, that asks for the co-occurrences of a word's lemma and POS or the sentences that exemplify a co-occurrence, several SQL queries are made in the database to get the information about that word or that co-occurrence. This information is organized in a JSON object and sent to the client.

In the client-side, the code was developed from the AngularJS[7] framework that allows the use of HTML and CSS, as declarative and presentation languages respectively.

To begin the process, the user is presented a form where he/she can choose the word's lemma and POS and also the association measure. However, the user can still choose some additional options like the minimum frequency of each co-occurrence and the maximum number of words in each dependency-property pattern. By default, these values are set two and ten, respectively. The collected information is organized in a JSON object and sent to the server. When the searched word does not exist or there are no results for it, the user is informed.

When the word has a result, this is presented in two groups: word to the left and word to the right with whom it co-occurs. For each dependency-property

[7] `https://angularjs.org` (last visit 26/09/2015).

pattern in which the target word is involved, the words that co-occur with the target word are presented in descending order of the selected association measure value. Each word that co-occurs is displayed in the format *lemma* $(l : m)$, where l is the base 2 logarithm of the co-occurrence's frequency, and m the value of the selected measure. The user may change the association measure without having to re-entering the information about the current word.

Fig. 5. Co-occurrences for a noun `país` ordered by the LogDice measure.

Figure 5 shows an example with a set of co-occurrences detected for the noun *país*. The association measure selected for this result was the*LogDice*.

When the target word is a verb, the result shows subjects, direct complements, essential complements, preposicional complements and the adverbs that modify the verb. In the case of the adjective, the result shows the adverbs that modify it and the nouns that it modify. For the adverb, the result shows the other adverbs that modify it and the adjectives, nouns, verbs, and other adverbs that the search adverb modify. For this, it also shown how many cases where the adverb modifies the entire sentence.

In the result, it is possible to ask for more information about a specific co-occurrence. By clicking on a word present in the result, the server is queried for a set of sentences in which this word occurred. As a result the user is presented the details of this co-occurrence and the set of sentences that exemplify the co-occurrence in the corpus.

4 Evaluation

In the evaluation of this system, the CETEMPúblico [22] corpus processed by STRING was used. The processed corpus occupies 237 GB, divided into 20 parts with 12 GB each. Each part has 210 XML files with 60 MB each. The extraction tool was executed in a x86_64 Unix machine, with 16 CPUs Intel(R) Xeon(R)E5530 2.40GHz and 48 390 MB of memory.

The extraction tool for a single file (`Parte01aaa.xml` with 60.8 MB) from the corpus was executed in 32.5 seconds, where 9.76 seconds were consumed by XIPAPI to import the XML file,10.65 seconds by the co-occurrence extraction and 11.69 seconds to store this and the sentences in the database, that after the execution occupies 4.33 MB. In this times, just the time to store the information will increase, because the database's response time will increase.

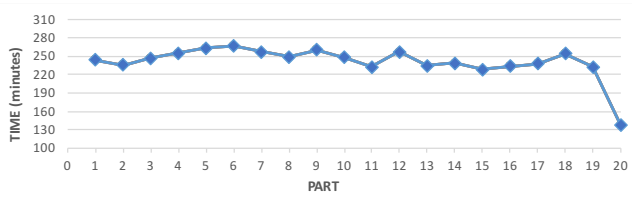

Fig. 6. Time consumed by each part of the CETEMPúblico corpus.

Figure 6 shows the performance of the tool, presenting the time spent on each part of the corpus. The tool processed the corpus in 4,878 minutes (3 days, 9 hours and 18 minutes), which corresponds to, in an average, 243 minutes (4 hours and 3 minutes) by each part of the corpus and 1 minute e 9.4 seconds by file, thus this increased 36.9 seconds. In the chart, it can be seen that the execution time is constant along the corpus, except in part 20, because this has a smaller size. The association measures were calculated in 146 minutes (2 hours and 26 minutes).

The evolution of the database during the process is not linear. Initially, the growth is more pronounced but tends to decrease as the process evolves. This happens because in the begin, the database is empty and more information is stored. At the end of the process, the database has 6.8 GB. With the association measure calculated, this has 7.3 GB (association measures occupy 0.5 GB).

The chart from Figure 7 shows the distribution of information stored in a database table. The tables `Dependency`, `Property` and `Corpus` do not appear on the chart because together they represent only 6 KB in the database. The tables `Sentences` and `Exemplifies` represent 76% of the total size of the database. This space is not larger because it was decided to store only fifteen sentences for each co-occurrence pattern. The others, that store the co-occurrence information occupy just 1.75 GB.

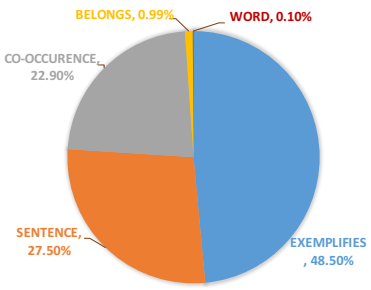

Fig. 7. Information distribution per table in the database.

In the extraction of CETEMPúblico 308,573 different word's lemmas were collected, and 11,244,852 different co-occurrences patterns that occurred 51,321,751 times in the corpus.

The evaluation of the web application consisted in measuring the time that the application takes to show the results for a word. This time consist mainly in the execution time of server-side, there is, the time to run the different SQL queries for each word's POS. To evaluate this two lemmas were used for each word's POS, one with high and another with low frequency.

Table 1 shows the results for the web application performance. The time for the word's lemma with greater frequency is slightly lower than the word's lemma with smaller frequency. In order to obtain a faster system response, it

Table 1. Time that the application takes to show the results to a word.

	Noun		Verb		Adverb		Adjective	
Lemma	caneta	país	otimizar	ser	nitidamente	não	lindo	grande
Frequency	998	196,140	972	2,533,385	964	1,362,286	997	263,518
Time (s)	0.102	0.111	0.103	0.760	0.037	0.174	0.067	0.256

was implemented indexes in the database's tables to reduce the time for each SQL query.

Table 2. Time that the application takes to show the sentences examples for a co-occurrence.

Co-occurence	**Frequency**	**Time (s)**
(centro,país)	789	0.024
(país,praticamente)	46	0.025
(país,populoso)	16	0.035

Table 2 shows the time that the application takes to show the sentences examples for a co-occurrence. It can be observed that the higher the frequency of co-occurrence, the lower is the waiting time. This can be explained due to size of the `Exemplifies` table, because for a co-occurrence with low frequency, it may need to iterate much of the table.

5 Conclusion

In this document the importance of the co-occurrence patterns for understanding the language was showed, as well as the need to provide easy access to this type of data, so that they can be analyzed by linguists or students of Portuguese.

Nowadays, there are some tools like DeepDict, SketchEngine and *Wortschatz* that provide information on the co-occurrence patterns of a word in Portuguese corpora. These tools are based on different natural language processing systems and adopt different measures of association.

With the processing produced by STRING, we intended to take advantage of the rich lexical resources in the system, as well as its sophisticated syntactic and semantic analysis in order to produce results that the above systems do not provide. Thus, a tool was created that extract co-occurrences patterns from a corpus processed by STRING, and then finds and stores the co-occurrences in a database. Next, for each co-occurrence stored, it calculates the different association measures.

The web application developed provides users with an interface that allows for the exploration of these co-occurrence patterns. This system makes possible to obtain a set of co-occurrences patterns for a target word, being possible to see detailed information about each co-occurrence within the sentences it occurred.

STRING rich lexical resources module a large number of compound entries (or multiword units/expressions), which as processed as much the same way as simple, single word units. It is therefore possible to search and analyse multiwords' distribution and co-occurrence patterns as well.

The results in the project's evaluation are acceptable, the run time of the extraction tool remains constant throughout the corpus, while the size of the database does not grow linearly, indicating that this is not repeating information. As for the web application response time, the results are good, that allows quickly queries to the information stored in the database.

In the future, it is necessary to increase the number of corpora present in the database to allow different results, making possible to compare the co-occurrence of a word for different corpora and create *thesauri* for each word from the stored information.

References

1. Aıt-Mokhtar, S., Chanod, J.P., Roux, C.: Robustness beyond shallowness: Incremental deep parsing. Natural Language Engineering 8, 121–144 (2002)
2. Bick, E.: The Parsing System "Palavras". Automatic Grammatical Analysis of Portuguese in a Constraint Grammar Framework. Aarhus University Press, Arhus (2000)
3. Bick, E.: DeepDict - A Graphical Corpus-based Dictionary of Word Relations. In: Proceedings of NODALIDA 2009. NEALT Proceedings Series. vol. 4, pp. 268–271. Tartu: Tartu University Library (2009)
4. Biemann, C., Bordag, S., Heyer, G., Quasthoff, U., Wolff, C.: Language-independent methods for compiling monolingual lexical data. In: Proceedings of CicLING. pp. 215–228. Springer (2004)
5. Carapinha, F.: Extração Automática de Conteúdos Documentais. Master's thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa (June 2013)
6. Chen, P.: The entity-relationship model—toward a unified view of data. ACM Transactions on Database Systems 1(1), 9–36 (1976)
7. Church, K., Hanks, P.: Word Association Norms, Mutual Information, and Lexicography. Computational Linguistics 16(1), 22–29 (1990)
8. Codd, E.: A relational model of data for large shared data banks. Commun. ACM 26(6), 64–69 (1983)
9. Dice, L.: Measures of the Amount of Ecologic Association Between Species. Ecology 26(3), 297–302 (Jul 1945)
10. Diniz, C., Mamede, N., Pereira, J.: RuDriCo2 - A Faster Disambiguator and Segmentation Modifier. In: INFORUM II. pp. 573–584 (September 2010)
11. Dunning, T.: Accurate methods for the statistics of surprise and coincidence. Computational Linguistics 19(1), 61–74 (1993)
12. Kilgarriff, A., Baisa, V., Bušta, J., Jakubíček, M., Kovář, V., Michelfeit, J., Rychly, P., Suchomel, V.: The Sketch Engine: ten years on. Lexicography 1(1), 7–36 (Jul 2014)
13. Kilgarriff, A., Rychly, P., Tugwell, D., Smrz, P.: The Sketch Engine. In: Proceedings of Euralex. vol. Demo Session, pp. 105–116. Lorient, France (Jul 2004)
14. Mamede, N., Baptista, J., Diniz, C., Cabarrão, V.: STRING: An Hybrid Statistical and Rule-Based Natural Language Processing Chain for Portuguese. In: PROPOR 2012. vol. Demo Session (April 2012)
15. Mamede, N., Baptista, J., Hagège, C.: Nomenclature of Chunks and Dependencies in Portuguese XIP Grammar 4.0. Tech. rep., L2F/INESC-ID, Lisboa (Maio 2013)
16. Manning, C., Schütze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge, MA, USA (1999)
17. Nobre, N.: Resolução de Expressões Anafóricas. Master's thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa (June 2011)
18. Quasthoff, U., Richter, M., Biemann, C.: Corpus Portal for Search in Monolingual Corpora. In: Proceedings of the 5th LREC. pp. 1799–1802 (2006)
19. Ribeiro, R.: Anotação Morfossintática Desambiguada do Português. Master's thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa. (March 2003)
20. Rychly, P.: Manatee/Bonito – A Modular Corpus Manager. In: P. Sojka, A. Horák (eds.) RASLAN 2008. pp. 65–70. Masaryk University, Brno (2007)
21. Rychly, P.: A Lexicographer-friendly Association Score. In: RASLAN 2008. pp. 6–9. Masarykova Univerzita, Brno (2008)

22. Santos, D., Rocha, P.: Evaluating CETEMPúblico, a Free Resource for Portuguese. In: Proceedings of the 39th Annual Meeting of ACL. pp. 450–457. ACL '01, Association for Computational Linguistics, Stroudsburg, PA, USA (2001)
23. Silberschatz, A., Korth, H., Sudarshan, S.: Database System Concepts. Connect, learn, succeed, McGraw-Hill Education (2010)
24. Smadja, F., McKeown, K., Hatzivassiloglou, V.: Translating collocations for bilingual lexicons: A statistical approach. Computational Linguistics 22(1), 1–38 (Mar 1996)
25. Vicente, A.: LexMan: um Segmentador e Analisador Morfológico com Transdutores. Master's thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa (June 2013)