# Assignment 2: Technical Report on the Pipeline

## Target Application: Searching for social action information in the Municipality of Paredes

**Advanced Information Extraction**

Vanessa Silva
(vanessa.silva@dcc.fc.up.pt)
*MAP-i 2018-2019*

### 1. Introduction

Information Extraction (IE) refers basically to the extraction of semantic information from the text. It includes tasks such as morphological analysis, named-entity recognition, resolution of co-referencing, etc.

Creating an IE application is not an easy task because it depends on several factors, such as, the domain/theme from which the application extracts information, language, specific cases/expressions, among others. For this it is recurrent to resort to the Natural Language Process (NLP) that studies the problems of the generation and automatic comprehension of natural human languages.

Within the scope of Advanced Information Extraction the goal is to develop a concrete IE application based on NLP. For this I will create an application whose purpose is to process and extract information about *social action in the Municipality of Paredes*, namely, information about places, contacts, objectives, requirements, actions, events, etc. Obtaining this type of information is very important given the relevance of the topic, the existence of large numbers of people in difficult situations in the Municipality, lack of dissemination of relevant information and difficulty in finding specific information.

This short technical report aims to report the main tasks developed for the implementation of the application, as well as describe the developed NLP pipeline.

### 2. Application Pipeline

Figure 1 illustrates the pipeline schema defined for the target application.

The orange color represent the tasks of collecting the unstructured information, pre-processing of the collected information and obtaining corpus to be processed, that is, the set of documents with information of interest. The green represent the NLP tasks, the yellow the ontology created, and the blue the task of information/knowledge extraction.

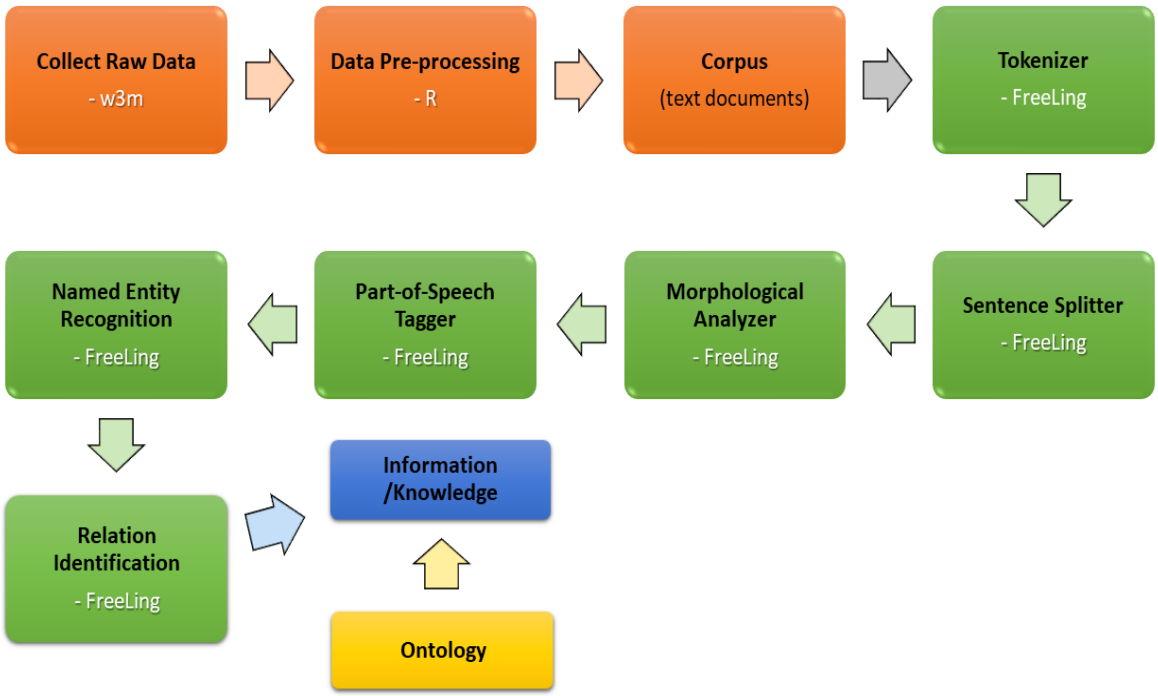


**Figure 1. Pipeline schema defined for the IE application.**

## 2.1. Corpus

A *corpus* is a large and structured collection of texts.

**2.1.1. Collect Raw Data.** The source for the documents is the web page of the Municipality of Paredes[1]. Initially it was also defined some online PDF files, however, so as not to make the corpus very heterogeneous, only the information of the web page was kept as source. All content is in portuguese language. An example can be seen in Figure 2.

To obtain the documents I have used the `w3m` software[2] which, compared to the `wget` software initially chosen, is more efficient for the desired return content. Because it is a pager/text-based web browser. For example, the meta-information of the web document structure will not be used (e.g., `<title></title>`), and the `w3m -dump` command automatically returns the information "clean" from this meta-information.

**2.1.2. Data Pre-processing.** To handle the raw data I used the software `R` where I deleted several lines of text before and several lines after the lines with the desired text, which referred to headers and footers of web pages; page titles and empty lines have been deleted; and text in list form (with items) has been converted to plain text.

[1] `https://www.cm-paredes.pt/pages/382`
[2] http://w3m.sourceforge.net/

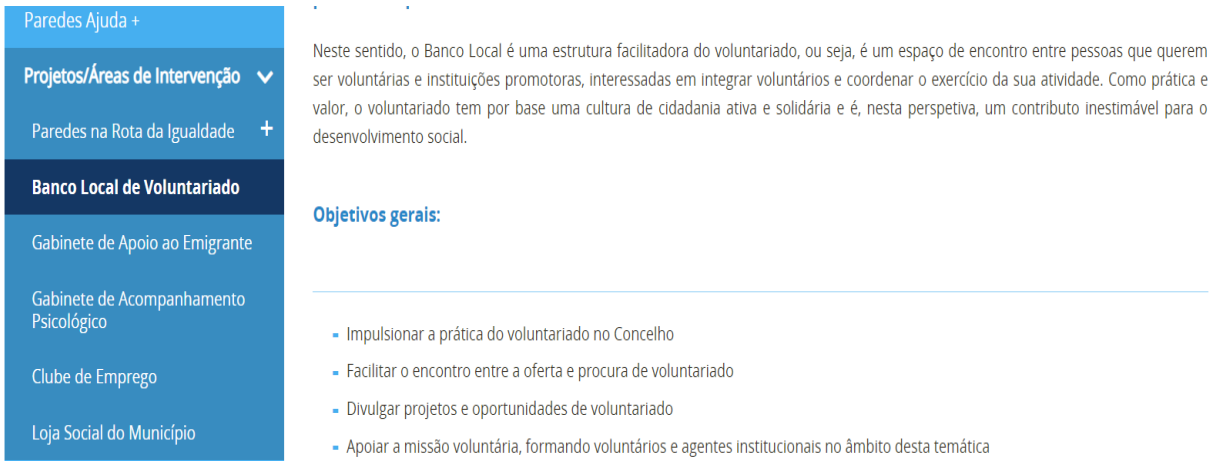


**Figure 2. Small sample of text to be extracted from the web page of the Municipality of Paredes.**

A small example of text obtained from the created Corpus:

```
Neste sentido, o Banco Local é uma estrutura facilitadora do
voluntariado, ou seja, é um espaço de encontro entre pessoas que
querem ser voluntárias e instituições promotoras, interessadas em
integrar voluntários e coordenar o exercício da sua atividade.  Como
prática e valor, o voluntariado tem por base uma cultura de cidadania
ativa e solidária e é, nesta perspetiva, um contributo inestimável
para o desenvolvimento social.  Objetivos gerais:  Impulsionar a
prática do voluntariado no Concelho; Facilitar o encontro entre a
oferta e procura de voluntariado; Divulgar projetos e oportunidades
de voluntariado; Apoiar a missão voluntária, formando voluntários e
agentes institucionais no âmbito desta temática.
```

### 2.2. Domain Independent

The domain-independent tasks that follow are NPL tasks that do not depend on the domain/theme of the text corpus.

For these tasks I exclusively used the *FreeLing* [1] (version 4.1) library which has an API for *Python* (programming language).

**2.2.1. Tokenizer.** FreeLing `tokenizer` converts plain text into a list of `word` objects corresponding to the tokens created.

Tokens obtained for the sample text above:

```
   [ [Neste]; [sentido]; [,]; [o]; [Banco]; [Local]; [é]; [uma];
[estrutura]; [facilitadora]; [do]; [voluntariado]; [,]; [ou]; [seja];
[,]; [é]; [um]; [espaço]; [de]; [encontro]; [entre]; [pessoas]; [que];
[querem]; [ser]; [voluntárias]; [e]; [instituições]; [promotoras]; [,];
[interessadas]; [em]; [integrar]; [voluntários]; [e]; [coordenar];
[o]; [exercício]; [da]; [sua]; [atividade]; [.]; [Como]; [prática];
[e]; [valor]; [,]; [o]; [voluntariado]; [tem]; [por]; [base]; [uma];
[cultura]; [de]; [cidadania]; [ativa]; [e]; [solidária]; [e]; [é];
[,]; [nesta]; [perspetiva]; [,]; [um]; [contributo]; [inestimável];
[para]; [o]; [desenvolvimento]; [social]; [.]; [Objetivos]; [gerais];
[:]; [Impulsionar]; [a]; [prática]; [do]; [voluntariado]; [no];
[Concelho]; [;]; [Facilitar]; [o]; [encontro]; [entre]; [a]; [oferta];
[e]; [procura]; [de]; [voluntariado]; [;]; [Divulgar]; [projetos];
[e]; [oportunidades]; [de]; [voluntariado]; [;]; [Apoiar]; [a];
[missão]; [voluntária]; [,]; [formando]; [voluntários]; [e]; [agentes];
[institucionais]; [no]; [âmbito]; [desta]; [temática]; [.]; ]
```

**2.2.2. Sentence Splitter.** FreeLing's sentence `splitter` receives lists of `word` objects (produced by the tokenizer) and buffers them until a sentence limit is detected, and returns a list of `sentence` objects. The sentence limits are: `.`, `?` and `!`.

Sentences obtained for the sample text above:

```
   [ Neste sentido, o Banco Local é uma estrutura facilitadora do
voluntariado, ou seja, é um espaço de encontro entre pessoas que querem
ser voluntárias e instituições promotoras, interessadas em integrar
voluntários e coordenar o exercício da sua atividade.  ]
   [ Como prática e valor, o voluntariado tem por base uma cultura
de cidadania ativa e solidária e é, nesta perspetiva, um contributo
inestimável para o desenvolvimento social.  ]
    [ Objetivos gerais:  Impulsionar a prática do voluntariado
no Concelho; Facilitar o encontro entre a oferta e procura de
voluntariado; Divulgar projetos e oportunidades de voluntariado; Apoiar
a missão voluntária, formando voluntários e agentes institucionais no
âmbito desta temática.  ]
```

**2.2.3. Morphological Analyzer.** Morphological analyzer in FreeLing is a meta-module that does not perform any processing itself, it calls submodules necessary for morphological analysis, such as: *punctuation detection*, *number detection*, *dates detection*, *dictionary search*, *multiword recognition*, *quantity recognition*, *affix checking* nad *named entity recognition*. This module (`morfo`) receives a `maco_options` object, containing information about which submodules need to be created and which configuration files should be used to create them. Each of these submodules receives a list of `sentence` and morphologically annotates each `word` of each sentence in the list.

To some of the configuration files (already provided by FreeLing) I have added infromation directed to the domain of my application, for example, `FreeLing/share/freeling/pt/locucions.dat` and `FreeLing/share/freeling/pt/chunker/grammar-chunk.dat`.

**2.2.4. Part-of-Speech Tagger.** FreeLing offers two PoS taggers with state-of-the-art accuracy (about 97%) [2], one HMM–based following [3] and another based on relaxation labelling [4]. For my application I chose to use the `hmm_tagger` class that implements a classical trigram Markovian tagger.

The `tagger` receives a list of `sentence` and disambiguates the PoS of each `word` in the given sentences. If I select analysis carries tokenization information, the word may be split in two or more new words.

The tags obtained for one part of the first sentence above can be seen in Table 1.

Note that in this stage the morphological analysis has already been performed, including NER (which we will explain below), and so we can check some interesting things, for example:

- The string `Banco Local` is a multiword `Banco_Local` and is a name (`N`) common (`C`) masculine (`M`) singular (`S`), but only appears as proper name (NP) because of having been re-tagged by the NER;
- the word `querem` is an verb (`V`) main (`M`) indicative (`I`) present (`P`) in the third person (`3`) of plural (`P`), and it refer to the verb `querer`;
- the word `Neste` is decomposed into words `Em` and `este`.

### 2.3. Domain Specific

The domain specific tasks that follow are NPL tasks that do depend on the domain and context of the text corpus.

For these tasks I also exclusively used the `FreeLing` library which has an API for `Python`.

**2.3.1. Named Entity Recognition and Classification.** The Named Entity recognition task is performed as a combination of local classifiers which test simple decisions on each word in the text [5].

The Freeling offers two different modules capable of performing the recognition of NE. Basic NER module which basically detects word sequences in capital letters, taking into account some functional words (for example, `Município de Paredes`) and capitalization at the beginning of the sentence. And the BIO NER module, which is based on machine learning, uses a classification algorithm (*AdaBoost* classifier) to decide whether each word is in a begin (B), inside (I) or outside (O).

I chose to use the last module because theoretically it is much better and after some testing this was verified in the whole. This module assigns the tag `NP00000` to all found entities and it is necessary to later apply another module of FreeLing, the Named Entity Classification, for the subsequent assignment of the correct tags, that is, the tag is changed to the label defined for entities recognized by FreeLing: **Person** (tag `NP00SP0`), **Geographical** location (tag `NP00G00`), **Organization** (`NP00O00`) and **Others** (tag `NP00V00`).

**Table 1. Word form, lemma and tag obtained for the content of the first sentence of example.**

| WordForm | Lemma | Tag |
|---|---|---|
| Em | em | SP |
| este | este | DD0MS0 |
| sentido | sentido | NCMS000 |
| , | , | Fc |
| o | o | DA0MS0 |
| Banco_Local | banco_local | NP00000 |
| é | ser | VMIP3S0 |
| uma | um | DI0FS0 |
| estrutura | estrutura | NCFS000 |
| facilitadora | facilitador | AQ0FS00 |
| de | de | SP |
| o | o | DA0MS0 |
| voluntariado | voluntariado | NCMS000 |
| , | , | Fc |
| ou_seja | ou_seja | RG |
| , | , | Fc |
| é | ser | VMIP3S0 |
| um | um | DI0MS0 |
| espaço | espaço | NCMS000 |
| de | de | SP |
| encontro | encontro | NCMS000 |
| entre | entre | SP |
| pessoas | pessoa | NCFP000 |
| que | que | PR0CN00 |
| querem | querer | VMIP3P0 |
| ser | ser | VMN0000 |
| voluntárias | voluntário | AQ0FP00 |

It is a Machine-Learning based module, so the classes can be anything the model has been trained to recognize. The model learns to rely in the **gazetteer** information, and performance will degrade if the analyzed corpus contains many unknown entities. Fortunately, this model allowed me to adapt the classifier to my application domain, by enriching the gazetteer.

The gazetters are stored in the files `FreeLing/share/freeling/pt/nerc/data/gazXXX-[cp].dat`, where XXX denotes the class of the lemmas contained in the file (PER, LOC, ORG, MISC), and the suffix "-p" or "-c" denotes whether the name is a complete NE ("-c") or a partial NE ("-p").

To this gazetters we add the entities of our domain as follows: just add lowercased names of complete entities to `gazXXX-c.dat`, and words that are part of multiword entities to `gazXXX-p.dat`.

The NE `classifier` receives a list of `sentence` and classifies all `word` tagged as proper nouns (NP) in the given sentences.

The entities obtained the text example above can be seen in the first two rows of the Table 2. The remaining entities are some examples of entities obtained for the whole corpus.

**Table 2. Entities obtained the text example.**

| WordForm | Lemma | Entity |
|---|---|---|
| Banco_Local | banco_local | NP00O00 |
| Concelho | concelho | NP00G00 |
| Município_de_Paredes | município_de_paredes | NP00G00 |
| Carta_Europeia_para_a_Igualdade_de_os_Homens | carta_europeia_para_a_igualdade_de_os_homens | NP00V00 |
| Europa | europa | NP00G00 |
| Comissão_para_Cidadania_e_Igualdade_de_Género | comissão_para_cidadania_e_igualdade_de_género | NP00O00 |
| POPH | poph | NP00O00 |
| Diagnóstico | diagnóstico | NP00V00 |
| Governo | governo | NP00O00 |
| Assuntos_Parlamentares | assuntos_parlamentares | NP00V00 |
| Teresa_Morais | teresa_morais | NP00SP0 |
| Protoloco | protoloco | NP00V00 |
| Paulo_Simões_Júlio | paulo_simões_júlio | NP00SP0 |

**2.3.2. Statistical Dependency Parser.** FreeLing has a statistical dependency analysis module, this is based on *Treeler*[3] machine learning library.

The `dep_treeler` receives a list of parsed `sentence` objects and associates to each of them a `dep_tree` object. The `dep_tree` is an n-ary tree where each node contains a reference to a node in a parse_tree. The structure of `dep_tree` establishes syntactic dependency relations between the constituents of the `sentence`.

The statistical dependence obtained for the example of the text above (for the first sentence) is:

```
    .
1 Em em SP - - - - - 3 case - -
2 este este DD0MS0 - - - - - 3 nmod - -
3 sentido sentido NCMS000 - - - - - 9 nmod - -
4 , , Fc - - - - - 9 mark - -
5 o o DA0MS0 - - - - - 6 det - -
6 Banco_Local banco_local NP00O00 - - B-ORG - - 9 nsubj - -
7 é ser VMIP3S0 - - - - - 9 cop - -
8 uma um DI0FS0 - - - - - 9 dep - -
9 estrutura estrutura NCFS000 - - - - - 19 nsubj - -
```

[3] http://devel.cpl.upc.edu/treeler

```
10 facilitadora facilitador AQ0FS00 - - - - - 9 xcomp - -
11 de de SP - - - - - 13 case - -
12 o o DA0MS0 - - - - - 13 det - -
13 voluntariado voluntariado NCMS000 - - - - - 9 nmod - -
14 , , Fc - - - - - 19 nsubj - -
15 ou_seja ou_seja RG - - - - - 19 advmod - -
16 , , Fc - - - - - 19 nsubj - -
17 é ser VMIP3S0 - - - - - 19 cop - -
18 um um DI0MS0 - - - - - 17 xcomp - -
19 espaço espaço NCMS000 - - - - - 0 root - -
20 de de SP - - - - - 21 case - -
21 encontro encontro NCMS000 - - - - - 19 nmod - -
22 entre entre SP - - - - - 23 case - -
23 pessoas pessoa NCFP000 - - - - - 19 nmod - -
24 que que PR0CN00 - - - - - 25 nsubj - -
25 querem querer VMIP3P0 - - - - - 23 acl:relcl - -
26 ser ser VMN0000 - - - - - 25 xcomp - -
27 voluntárias voluntário AQ0FP00 - - - - - 25 dep - -
28 e e CC - - - - - 27 cc - -
29 instituições instituição NCFP000 - - - - - 27 conj - -
30 promotoras promotor AQ0FP00 - - - - - 29 amod - -
31 , , Fc - - - - - 29 conj - -
32 interessadas interessar VMP00PF - - - - - 29 acl - -
33 em em SP - - - - - 34 mark - -
34 integrar integrar VMN0000 - - - - - 32 advcl - -
35 voluntários voluntário NCMP000 - - - - - 34 dobj - -
36 e e CC - - - - - 34 cc - -
37 coordenar coordenar VMN0000 - - - - - 34 conj - -
38 o o DA0MS0 - - - - - 39 det - -
39 exercício exercício NCMS000 - - - - - 37 dobj - -
40 de de SP - - - - - 43 case - -
41 a o DA0FS0 - - - - - 43 det - -
42 sua seu DP3FSS - - - - - 43 dep - -
43 atividade atividade NCFS000 - - - - - 34 ccomp - -
44 .  .  Fp - - - - - 27 mark - -
```

### 2.4. Ontology

An ontology is a formal description of concepts and relationships that can exist for a community of human and/or machine agents. The notion of ontologies is crucial for the purpose of enabling knowledge sharing and reuse [6].

To create the ontology of my application I used Protégé[4]. This ontology can be seen in Figure 3.

[4]https://protege.stanford.edu/

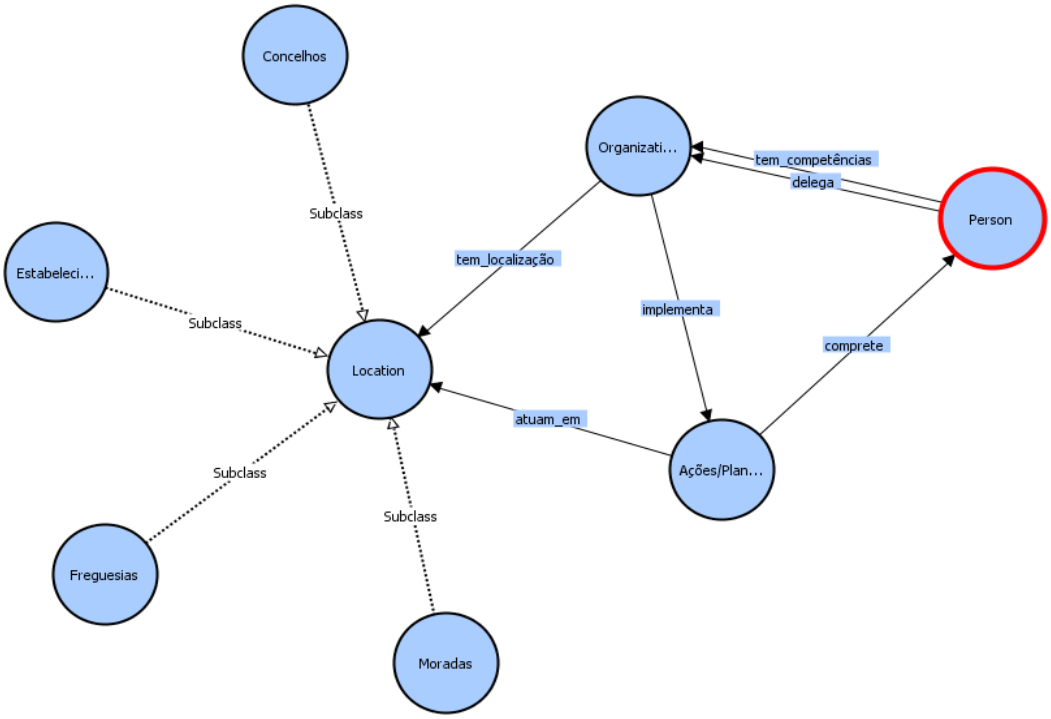


**Figure 3. Ontology schema defined for the application target.**

## 3. Information/Knowledge

Unfortunately, it was not possible to complete the application, that is, I could not relate the triples and the ontology for application in `Jena`. With mentioned in the application and after coming back, these days, trying to complete the application did not really succeed.

Throughout the work I came across several problems. I initially had several problems building the FreeLing API for `python` on Windows operating systems, where I later found that it is a problem that has already been reported, so I had to build on Linux. I also explored several of the functions of FreeLing and tried to understand the different implementations and differences for the different languages. Other tasks involved changing some configuration files to fit the application domain and training some of the models for more accurate results.

## References


[1] L. Padró and E. Stanilovsky, "Freeling 3.0: Towards wider multilinguality," in *Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*. Istanbul, Turkey: ELRA, May 2012.

[2] L. Padró, M. Collado, S. Reese, M. Lloberes, and I. Castellón, "Freeling 2.1: Five years of open-source language processing tools," in *7th International Conference on Language Resources and Evaluation*, 2010.

[3] T. Brants, "Tnt: a statistical part-of-speech tagger," in *Proceedings of the sixth conference on Applied natural language processing*. Association for Computational Linguistics, 2000, pp. 224–231.

[4] L. Padró, "A hybrid environment for syntax-semantic tagging," *arXiv preprint cmp-lg/9802002*, 1998.

[5] X. Carreras, L. Màrquez, and L. Padró, "A simple named entity extractor using adaboost," in *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*. Association for Computational Linguistics, 2003, pp. 152–155.

[6] N. Guarino, D. Oberle, and S. Staab, "What is an ontology?" in *Handbook on ontologies*. Springer, 2009, pp. 1–17.