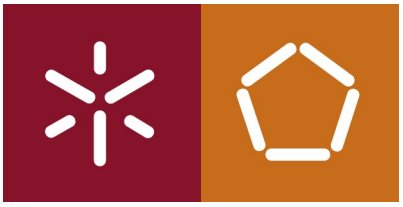

Universidade do Minho

Departamento Informático

Introdução a Processamento de Linguagem Natural

# Ontology Manipulation

João Costeira (a78073)
Paulo Mendes (a78203)
Bernardo Silva (a77230)

24 de Novembro de 2019

# Conteúdo

**1 Introdução** **2**

**2 Ontologia** **2**

**3 Descrição da Ferramenta** **3**

**4 Instalação** **3**

**5 Caso de estudo** **3**
5.1 Relações . . . . . 3
5.2 Código . . . . . 4
5.3 Treino . . . . . 4
5.3.1 Parser . . . . . 4
5.4 Graphics . . . . . 5
5.5 Visualização . . . . . 5

**6 Conclusão** **8**

# 1 Introdução

No contexto do segundo trabalho da unidade curricula *IPLN*, os professores propuseram um tema para cada grupo com o objetivo de efetuar um trabalho com duas componentes: por um lado uma pesquisa sobre o tema proposto e por outro lado realizar um trabalho aplicacional para demonstrar uma utilização da ferramenta.
O tema proposto para o nosso grupo foi ***Ontology manipulation*** e utilizar a biblioteca ***Pronto*** para *Python* na componente prática.

# 2 Ontologia

O termo ontologia teve origem da palavra grega *-logia* (falar) *onto-* (ser). Tradicionalmente esta palavra é utilizada em disciplinas filosóficas, mas o termo foi adotado pela ciência computacional.
Em informática, ontologia é um modelo de especificação e conceptualização do conhecimento.
A ontologia permite a contextualização dos dados em termos informáticos. Em vez de efectuar a representação de dados isoladamente numa dado formato, as ontologias encontram-se orientadas à relação. Assim a associação de termos é possível como acontece com o cérebro humano, por exemplo ao pensar numa palavra existir um conjunto de outras palavras associadas e a questão de ambiguidade, possíveis diferentes significados de acordo com a contextualização.
As classes permitem representar elementos e por sua vez relações as interacções existentes entre elas.
Graficamente ontologias podem ser representadas com duas componentes:

- Oval – para representar classes
- Setas – representar as relações.

Um exemplo simples da relação entre diferentes instâncias de veículos e da associação com transporte públicos.

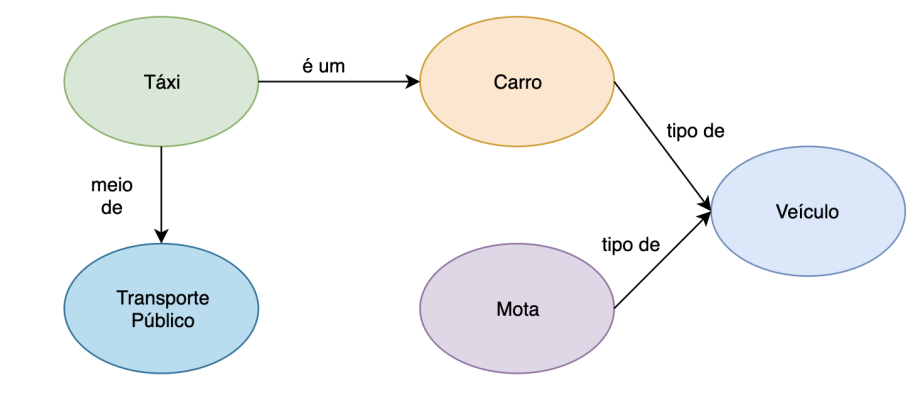


Fig. 1: Exemplo gráfico de uma ontologia.

Existe um conjunto de linguagens que propoem a representação de ontologias, por exemplo *Rdf(Resource Descrition Framework)* e *OWL(Web Ontology Language)*.

# 3 Descrição da Ferramenta

O *Pronto* [4] é um pacote de *Python* desenvolvido para trabalhar com ontologias. Geralmente esta ferramenta permite:

- Carregar um conjunto de formatos/ontologias:
  - *OBO(Open Biomedical Ontologies)*
  - *OBO Graphs*
  - *OWL(Web Ontology Language)*
  - *Ntologies*
- Permite efectuar o processamento nas ontologias e no fim exportá-las:
  - *OBO*
  - *OBO Graphs*
  - *json*

# 4 Instalação

Ferramentas necessárias instalar de modo a poder executar o programa.

- `pip3 install pronto`
- `pip3 install nltk`
- `pip install pyvis`

# 5 Caso de estudo

No contexto da unidade curricular *IPLN*, o caso de estudo escolhido foi efectuar o processamento de texto a fim de evidenciar as relações entre os elementos contidos no ficheiro.
Neste relatório encontra-se exemplos de execução deste programa sobre Os Maias de Eça de Queiroz.

## 5.1 Relações

Sobre o texto de entrada é necessário efectuar um processamento a fim de filtrar a informação necessária, as relações.
O critério de utilizado para filtrar as relações foi o desenvolvimento de dois *n-grams*.
O *trigrama* evidencia as relações, onde a palavra central é um verbo, porque normalmente os verbos representam relações/interacções entre agentes de numa frase.
Por outro lado, é necessário *bigramas* para representar relações directas entre duas palavras, por exemplo encontrar dois nomes seguidos numa frase. Caso contrario, informação que contem sequências de nomes, por exemplo um nome própria, seria ignorada.

```
#Trigrams
(Palavra,Verbo,Palavra)
#Bigrams
(Palavra,Palavra)
```

A seguinte imagem evidencia um exemplo de relações com trigramas (com verbos) ou de bigramas (sem verbos, relação direta).


Fig. 2: N-Grams utilizados neste trabalho.

## 5.2 Código

O código deste trabalho encontra-se separado em três secções, *treino.py*, *parser.py* e *graphics.py*.

## 5.3 Treino

A *script* *treino.py* é responsável por gerar o ficheiro *mac_morpho.pkl*. O objetivo deste código é treinar a rede de modo a poder identificar gramática portuguesa, neste trabalho o essencial é identificar os nomes e verbos para representar as relações.
Após a geração do ficheiro *mac_morpho.pkl*, não é necessaria a execução desta *script* novamente, porque o treino gramatical já foi efetuado.

### 5.3.1 Parser

A *script parser* é fundamental para guardar em ficheiro todas as relações existentes num texto.

Com a função `getALLFile()` é obtido o corpo de todos os ficheiros da diretoria atual ou com a função `getFromOneFile(path,fname)` abrir apenas um ficheiro especifico.

Numa fase seguinte, o corpo de um ou mais ficheiros é armazenado numa matriz, onde cada linha é uma frase do corpo do texto de entrada`getSentenses(corpus)` e cada elemento da coluna é uma palavra da frase.

Sobre cada palavra da matriz, é efectuada o *map* de modo a gerar uma matrix de pares: *(Palavra,Gramática)*. Sobre estes pares, é efectuada a filtragem das palavras classificadas como nomes, nomes próprios e verbos.

Após a geração da matriz filtrada, os *bigramas* e *trigramas* são gerados de acordo com dois tipos de sequencias no texto:

- `bigramas` - Geração de pares de palavras (nomes ou nomes próprios) consecutivos no texto.
- `trigramas` - Capturar sequências relacionais no texto que possuam o seguinte padrão: Palavra→Verbo→Palavra , onde a palavra pode ser um nome ou um nome próprio

O conjunto das relações são armazenadas em ficheiros *.csv*. Cada ficheiro *csv* possui três colunas: a *Source*, a Target e a *Weight*.
*Source* e Target são as palavras que estão relacionadas diretamente (no caso dos trigramas a relação (P1,V,P2) é separada em (P1,V),(V,P2)), o *Weight* é o peso da ligação do grafo, ou seja, o número de vezes que esse par de palavras se relacionou no texto na orientação (*source*→*target*).

## 5.4 Graphics

A *script graphics.py* é responsável por a partir de um ficheiro .csv gerar um ficheiro *.html* que permite a visualização do grafo.
Os grafos são representados com a biblioteca *Network*.

```
from pyvis.network import Network #graphics
import pandas as pd #csv
```

## 5.5 Visualização

No ficheiro *nomes.html* encontram-se representada todas as relações entre duas palavras consecutivas no texto e no ficheiro *verbs.html* todas as relações palavra→verbo→palavra.

A seguinte imagem contem todos os pares de palavras que se encontram relacionadas por intermédio de um dado verbo.

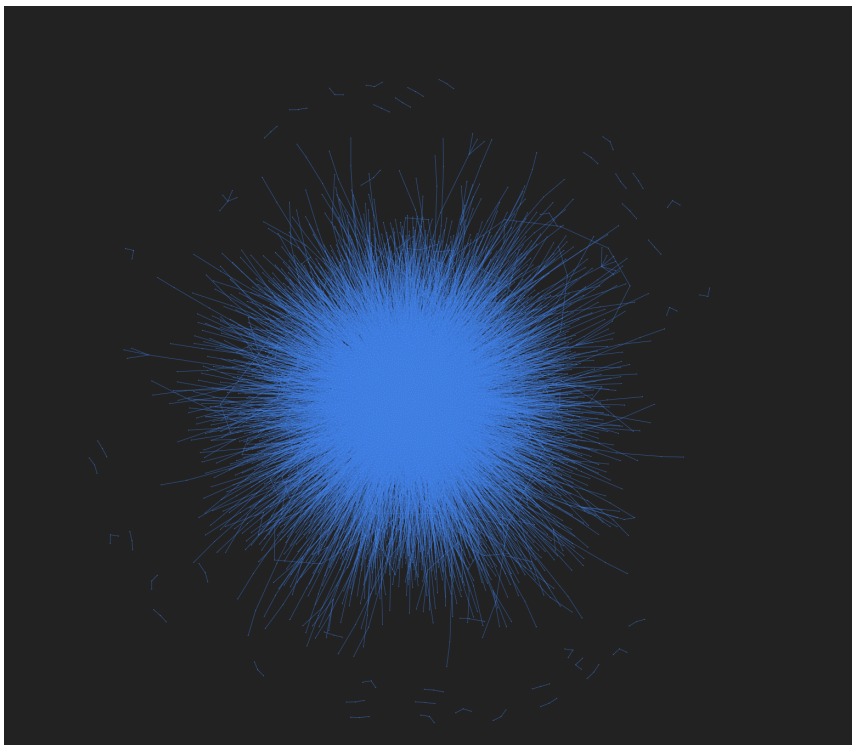

Fig. 3: Todas as relações existentes entre duas palavras por intermédio de um verbo na obra Os Mais de Eça de Queiroz

Devido à magnitude da obra, filtramos parte da informação do grafo. A seguinte imagem contem todas as relações que a Maria possui na obra.

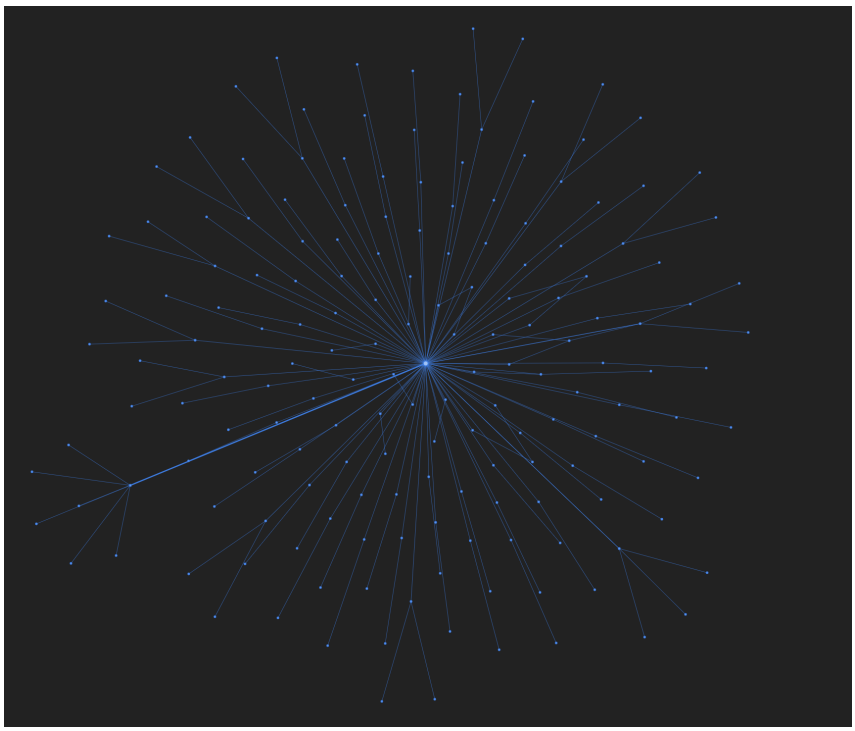

Fig. 4: Grafo com todas as relações com origem na personagem Maria

Ao ampliar a imagem, conseguimos observar que o nodo central do grafo é a Maria, pois todas as relações partem desta personagem.
Cada aresta vai ligar o nodo Maria a um verbo, nodo que representa a relação entre duas

entidades. A espessura da aresta é proporcional ao número de ocorrências do par de nodos que formam essa aresta.
A seguinte imagem contem o nodo central e que relações possui com as outras palavras.

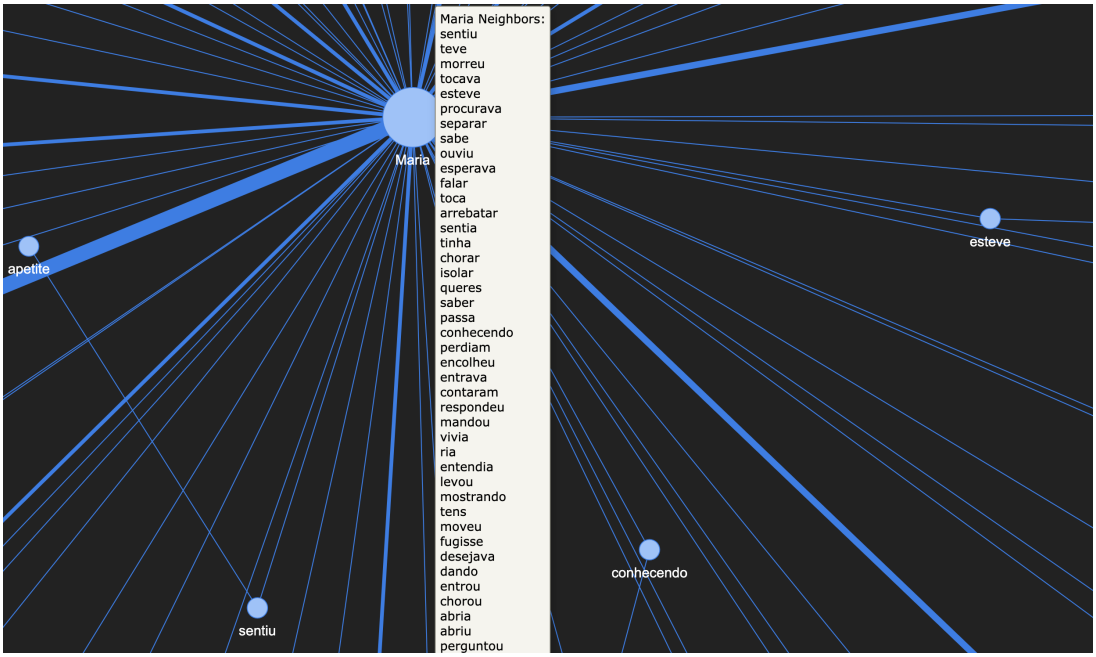


Fig. 5: Relações da personagem Maria

No outro lado, o nodo verbo vai relacionar com a outra palavra que está no extremo da relação.
A seguinte imagem contem as palavras com que o qual a Maria se relaciona com a palavra era, frase na semelhantes a "a Maria era ..."ou "Maria era uma ....":

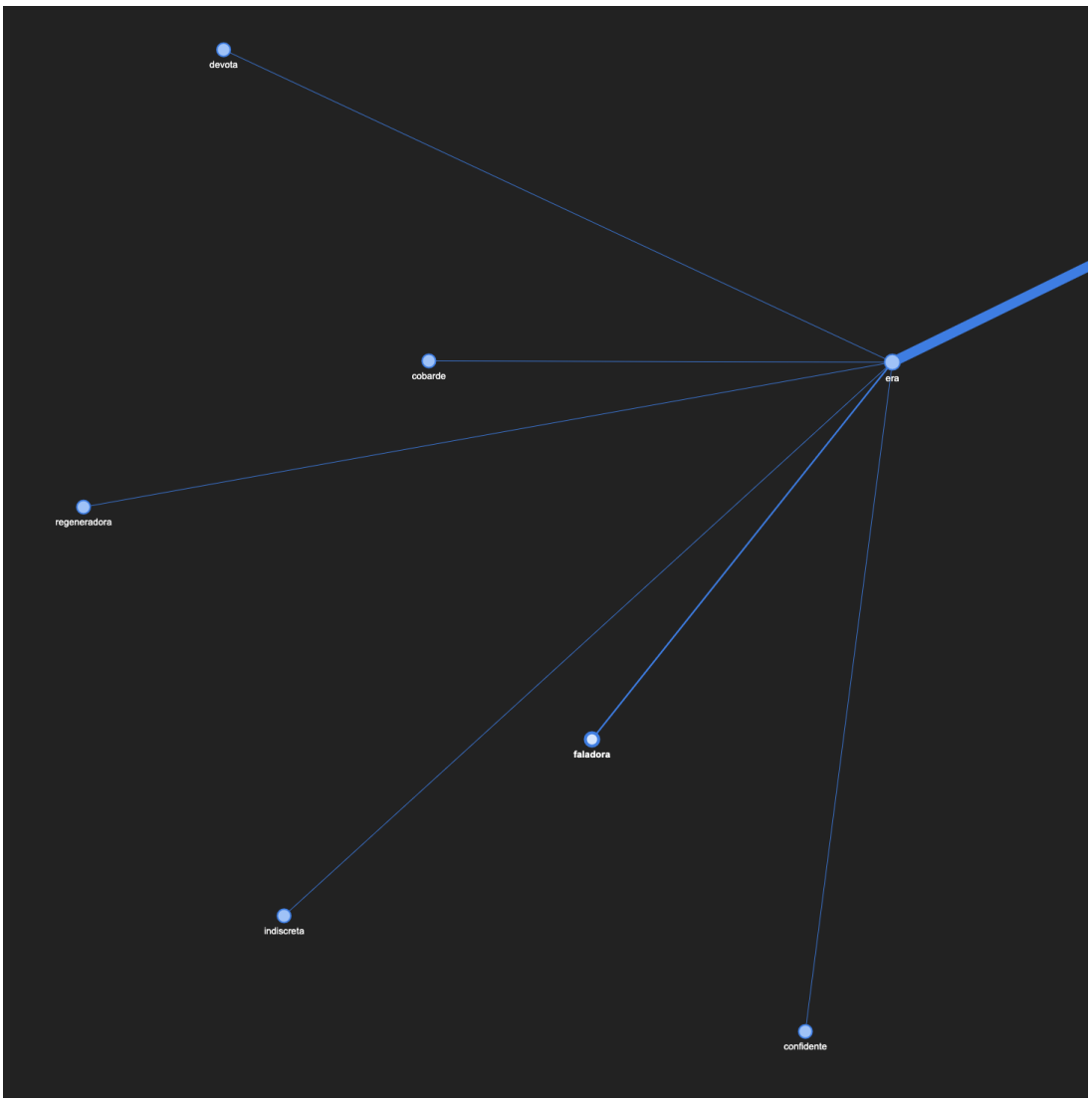


Fig. 6: Maria relacionar com o verbo era

# 6 Conclusão

Com este trabalho foi possível desenvolver os conhecimentos de processamento de linguagens de forma a identificar elementos gramaticais e processar-los no sentido de gerar ficheiros padronizados (*csv* ou *json*) com essa informação.
A grande dificuldade encontrada no trabalho foi a implementação da biblioteca pronto. De acordo com a pesquisa efectuada, essa ferramenta encontra-se especializada para ser utilizada em áreas associadas à biologia. À medida que o projecto foi desenvolvido orientado à unidade curricular, o trabalho obtido afastou-se da área da biblioteca pronto. Assim o trabalho focou-se no processo de geração de ficheiros com palavras que se encontram relacionadas numa frase e por fim é gerado grafos de modo a visualizar relações existentes entre palavras numa obra.

# Referências


[1] https://en.wikipedia.org/wiki/Ontology_(information_science), Acedido em 15 de Novembro de 2019.

[2] http://www.math.ubbcluj.ro/~didactica/pdfs/2013/didmath2013-06.pdf, Acedido em 15 de Novembro de 2019.

[3] Ciminano,Philipp, Ontology Learning and Population from Text,acid-free paper,2006 https://link.springer.com/content/pdf/10.1007%2F978-0-387-39252-3.pdf, Acedido em 15 de Novembro de 2019.

[4] Larralde, Martin, Pronto, 10 Novembro de 2019, https://buildmedia.readthedocs.org/media/pdf/pronto/latest/pronto.pdf?fbclid=IwAR2Qb7BIEOkKP58JXYfDYkDeNG_hMNUl6VP8s-9gPoFfzIWgCvWUq-o5A30 , Acedido em 15 de Novembro de 2019.