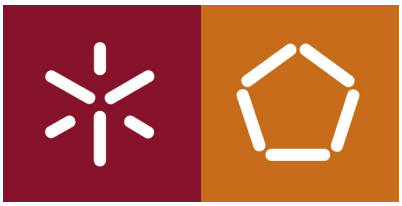

Universidade do Minho

Departamento de Informática

Scripting no Processamento de Linguagem Natural

Editor de letras

Francisco Oliveira (a78416)
Raul Vilas Boas (a79617)

1 de Julho de 2019

Conteúdo

1 Introdução **2**

2 Descrição da Ferramenta **3**
2.1 Palavras relacionadas 3
2.2 Rimas 4

3 Funcionalidades **5**

4 Conclusão **8**

1 Introdução

Para este trabalho prático foram apresentadas várias opções das quais escolhemos a opção 5.

Nesta opção o objetivo era, usando a metodologia de *Pat Pattinson* ou similar, construir uma lista de palavras relacionadas e de rimas para uma dada palavra. Pretendia-se também que fosse utilizado o *WordNet* ou fonte similar para obter as palavras relacionadas e rimas.

Com este objetivo desenvolvemos um *script* Python capaz de receber um input de palavras, aceitar algumas opções para alterar alguns parâmetros de execução e finalmente apresentar o resultado.

Nos próximos capítulos vamos descrever a ferramenta em maior detalhe, referindo as funcionalidades desenvolvidas e como a utilizar.

2 Descrição da Ferramenta

O processo de criação de músicas pode ser considerado impossível ou muito trabalhoso para alguns pois pode requer criatividade e sorte. Desta forma criamos um ferramenta com o objetivo de auxiliar o utilizador na criação de músicas seguindo a metodologia de *Pat Pattison*. Esta metodologia consiste em com base num tema para a música escolhida arranjar palavras que estejam relacionadas à palavra do tema escolhido, depois com essas podemos procurar outras palavras relacionadas com essas e guardar as mais relevantes. A partir desta lista das palavras basta arranjar rimas dessas mesmas palavras e depois temos um conjunto de palavras que podemos usar para auxiliar na construção da música tornado o processo mais fácil. Deste modo, a nossa ferramenta consiste no fornecimento de várias palavras pelo utilizador em que o programa depois vais buscar palavras relacionadas a essas e depois rimas dessas palavras.

Desta forma, a ferramenta está dividida em 2 partes, uma que consiste na busca de palavras relacionadas e outra que consiste na procura de rimas.

2.1 Palavras relacionadas

Para se conseguir arranjar as palavras relacionadas utilizou-se o *website* lexico.pt que fornece esta informação como observado na figura 1.

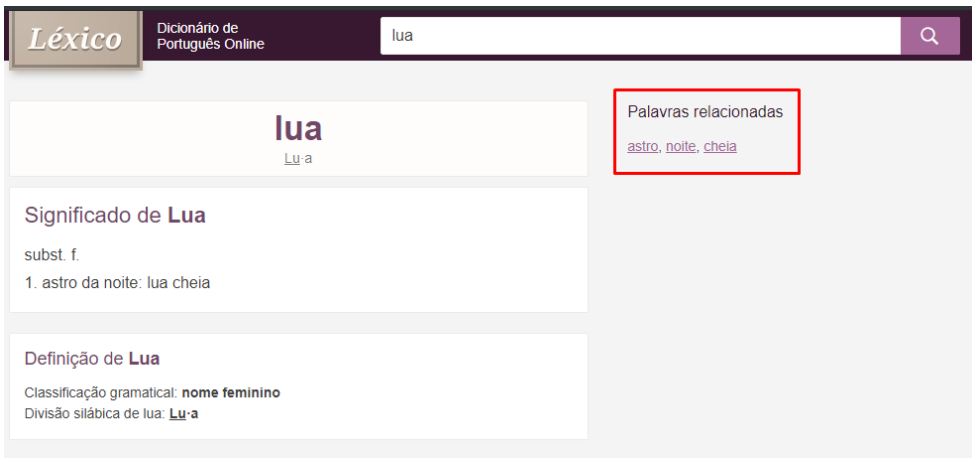

Fig. 1: Página lexico.pt

Realizando uma inspeção na página verificou-se que essas palavras estavam contidas no HTML da página (figura 2) e que era assim possível extrair o seu conteúdo.

```
▼<div id="relacionadas">
   <div class="title">Palavras relacionadas</div>
  ▼<div class="words">
     <a href="/astro/">astro</a>
     ", "
     <a href="/noite/">noite</a>
     ", "
     <a href="/cheia/">cheia</a>
   </div>
 </div>
```

Fig. 2: HTML da página lexico.pt

Para assim se conseguir obter essa informação para qualquer palavra fornecida pelo utilizou-se criou-se uma função que retorna o conteúdo HTML da página. Esta função

consiste em concatenar o URL base da página com a palavra e fazer um *request* a essa página. Isto vai retornar o HTML da página em que depois utilizando o *BeautifulSoup* fazemos *parse* do HTML.

```
def getHTML(word):
  urlBase = "https://www.lexico.pt/"
  word = unidecode.unidecode(word)
  composedURL = urlBase + word + "/"
  response = requests.get(composedURL).content
  soup = BS(response,'html.parser')
  return soup
```

Após obtermos o HTML tratamos de limitar o seu conteúdo, utilizando o *Beautifulsoup*, para assim, apenas obtermos as palavras pretendidas.

```
def get_content(soup):
  lista_soup = []
  soup = soup.find('div',{'class':'words'})
    if soup is not None:
      lista_soup = soup.find_all('a')
  return lista_soup
```

Assim, após se conseguir todas as palavras relacionadas que apareciam no *website* para uma determinada palavra criou-se um dicionário em que as *keys* são as palavras fornecidas pelo utilizador e os *values* a lista de palavras relacionadas. Segue-se o resultado obtido para as palavras lua, carro e mar.

```
{
 'lua': ['astro', 'noite', 'cheia'],
 'carro': ['automóvel', 'carruagem', 'coche', 'veículo', 'viatura'],
 'mar': ['grande', 'extensão', 'água', 'interior']
}
```

2.2 Rimas

Para se conseguir arranjar as rimas utilizou-se o *website* rhymit.com/pt que fornece esta informação como observado na figura 3.

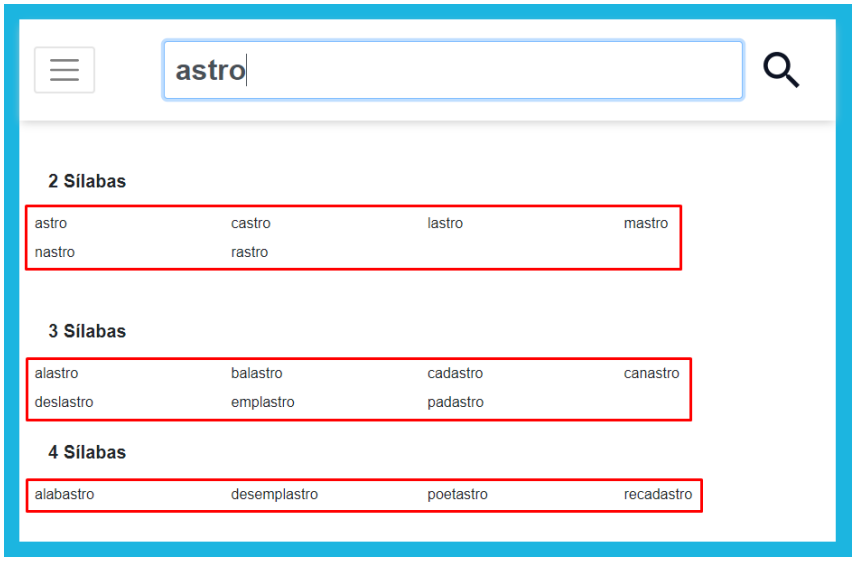

Fig. 3: Página rhymit.com/pt

Realizando uma inspeção na página verificou-se que as rimas estavam contidas no HTML da página (figura 4) e que era assim possível extrair o seu conteúdo.

Fig. 4: HTML da página rhymit.com/pt

Para assim se conseguir obter essa informação para qualquer palavra fornecida criou-se novamente uma função que retorna o conteúdo HTML da página seguindo a mesma estrutura da função para as palavras relacionadas.

```
def getHTML(palavra):
  urlBase = "https://www.rhymit.com/pt/palavras-que-rimam-com-"
  rhymeURL = urlBase + palavra
  response = requests.get(rhymeURL).content
  soup = BS(response,'html.parser')
  return soup
```

Após obtermos o HTML tratamos de limitar o seu conteúdo, utilizando o *Beautifulsoup*, para assim, apenas obtermos as palavras pretendidas.

```
def get_content(soup):
    lista_soup = soup.find_all('div',{"class":"syllableBlock"})
    return lista_soup
```

Assim, após se conseguir todas as rimas que apareciam no *website* para uma determinada palavra criou-se um dicionário em que as *keys* são as palavras fornecidas pelo utilizador e os *values* a lista de rimas. Segue-se o resultado obtido para as palavras astro e noite.

```
{
  'astro': ['alabastro', 'desemplastro', 'poetastro', 'recadastro', ... ],
  'noite': ['abiscoite', 'boa-noite', 'meia-noite', 'afoite', ... ]
}
```

3 Funcionalidades

Após ambos os *scripts* serem criados e ser possível obter toda a informação pretendida criou-se um *script* capaz de ao receber o input do utilizador e imprimir as palavras relacionadas dessas e depois a partir dessas o utilizador escolhe algumas delas para assim depois imprimir algumas rimas dessas palavras. Sendo que um possível *output* do programa é o seguinte.

```
Escreva algumas palavras.
lua carro mar

Relacionadas
```

```
lua: astro noite cheia
carro: automóvel carruagem veículo
mar: grande extensão água

Escolha algumas das palavras apresentadas.
astro noite carruagem

Rimas
astro: desemplastro alabastro mastro rastro cadastro
noite: abiscoite boa-noite amoite pernoite meia-noite
carruagem: vilanagem lavagem desempanagem bobagem betonagem
```

De modo a tornar o solução mais flexível criou-se algumas *flags* utilizando o package *argparse*.

- rel - Permite definir o número de palavras relacionadas a mostrar.
- rima - Permite definir o número de rimas a mostrar.
- input - Permite fornecer um ficheiro como *input*.
- output - Permite retornar o *output* para um ficheiro.

A titulo de exemplo, podemos correr a seguinte linha:

```
$ python3 main.py --rel 2 --rima 7 --input input.txt --output out.txt
```

Este comando usa a *flag* `--rel` para o programa devolver 2 palavras relacionas (em vez das *default* 3), a *flag* `--rima` para o programa devolver 7 rimas (em vez das *default* 5), a *flag* `--input` para receber o *input* apartir do ficheiro e não do *stdin* e a *flag* `--output` para enviar o output para o ficheiro em vez do *stdout*.

É importante referir que quando usamos a *flag* –input a segunda parte, onde se escolhem as palavras relacionas para as quais vamos apresentar as rimas, é ignorada, apresentando as rimas para todas as palavras relacionadas encontradas no passo anterior.

Os ficheiros de input e respetivo output são os seguintes:

```
lua carro mar
```

Listing 1: input.txt

```
lua carro mar

Relacionadas:
lua: astro noite
carro: automóvel carruagem
mar: grande extensão

Rimas:
lua: ablua intua institua rua preceitua anua acentua
astro: mastro poetastro castro padastro alabastro balastro lastro
noite: afoite amoite pernoite meia-noite boa-noite açoite abiscoite
carro: esbarro barro escarro tarro bizarro desamarro charro
automóvel: papamóvel móvel semimóvel eletromóvel telemóvel imóvel aeromó
    vel
carruagem: libertinagem pastagem tiragem caguetagem peonagem estucagem
    voragem
mar: brilhar quartar fanar brear ervar lacrar fretar
```

```
grande:  debande desmande mande sande ande expande abrande
extensão:  estação lousão fotão coirmão catalão nação jordão
```

Listing 2: out.txt

4 Conclusão

Neste trabalho prático desenvolvemos uma ferramenta capaz de facilitar a busca de palavras relacionadas e rimas para a criação de um musica, etc.

Com o trabalho desenvolvido, consideramos ter atingido os objetivos propostos e criado uma ferramenta de fácil uso, com algum grau de personalização através do uso de *flags*.

Como trabalho futuro poderia-se criar uma interface para dar ao utilizador a escolha entre o terminal e interface, e criar mais *flags* como potencialmente escolha de linguagem permitindo o uso da ferramenta por mais utilizadores.

Referências

[1] `https://www.lexico.pt` Acedido em 27 de Junho de 2019.

[2] `https://www.rhymit.com/pt` Acedido em 27 de Junho de 2019.