Scripting no Processamento de Lingua Natural

Teste, 2023-06-07

1 Word Embeddings

Assuma o ficheiro `termos.txt` que contém termos médicos (um por linha):

```
Fribrilhação auricular
Fémur
Artéria Aorta
tricipete
(...)
```

Assuma um outro ficheiro `categorias.txt` que contém um conjunto de categorias que podem ser utilizadas para classificar cada uma das entradas do ficheiro `termos.txt`

```
doença
ossos
artérias
vasos sanguíneos
músculos
anatomia
(...)
```

1. Implemente um programa que associe cada termo do ficheiro `termos.txt` a duas categorias do ficheiro `categorias.txt`. Para isso deve usar um modelo de word-embeddings para escolher as categorias semanticamente mais próximas de cada termo.

O output deve ser guardado num ficheiro `termos_cat.txt` com o seguinte formato:

```
tricipete@anatomia#músculos
femur@anatomia#ossos
Artéria Aorta@vasos sanguíneos#artérias
(...)
```

2. Dado o resultado do exercício anterior e um novo ficheiro `termos_2.txt` que contém novos termos médicos, implemente um programa que associe uma categoria a cada um dos novos termos, fazendo uso de analogias de word-embeddings.

2 Web Scrapping

Implemente um programa em Python que dado o URL correspondente ao website de um ginásio, extraia o nome, dificuldade, instrutor e horário de todas as modalidades desse ginásio.

Tome como exemplo as páginas HTML apresentadas abaixo.

Exemplo de página do plano de estudos: (https://www.ginasio.pt/modalidades)

```
<div class="container modalities">
    <h1 class="modalities-title">Modalidades no Ginásio</h1>
</div>
<div class="container modalities-list">
    <div class="modalidades-table">
        <table class="modalities-table">
            <tr>
                <th>Duração</th>
                <th>Modalidade</th>
```

```
                <th>Nível de dificuldade</th>
            </tr>
            <tr class="t-row-even">

                <td>120 minutos</td>
                <td> <a href="https://www.ginasio.pt/modalidades/ciclismo"> Ciclismo </a> </td>
                <td>Fácil</td>
            </tr>
            <tr class="t-row-odd t-row-special">

                <td>60 minutos</td>
                <td> <a href="https://www.ginasio.pt/modalidades/natacao"> Natação </a> </td>
                <td>Médio</td>
            </tr>

            (...)

        </table>
    </div>
```

Exemplo de página de uma Modalidade (https://www.ginasio.pt/modalidades/ciclismo)

```
<div class="container gym">
    <h1 class="gym-title">Ginásio da Universidade do Minho</h1>
</div>
<div class="container sport">
    <div class="modalidade-title">
        <h2><b>Ciclismo</b></h2>
    </div>
    <div class="modalidade-desc">
        <p class="sport-description">
            Ciclismo é uma atividade que envolve a repetição de um movimento ...
        </p>
        <ul class="sport-details">
            <li class="sport-detail">Modalidade: Ciclismo</li>
            <li class="sport-detail">Duração: 60 minutos</li>
            <li class="sport-detail">Nível de dificuldade: Médio</li>
            <li class="sport-detail">Instrutor: João Silva</li>
        </ul>
        <div class="sport-image">
            <img src="ciclismo.jpg" alt="Imagem de ciclismo">
        </div>
        <div class="sport-schedule">
            <h3 class="schedule-title">Horário:</h3>
            <table class="schedule-table">
                <tr>
                    <th>Dia</th>
                    <th>Horário</th>
                </tr>
                <tr>
                    <td>Segunda-Sexta</td>
                    <td>08:00 - 19:00</td>
                </tr>
                <tr>
                    <td>Sábado-Domingo</td>
                    <td>09:00 - 12:00</td>
                </tr>
            </table>
        </div>
    </div>
</div>
```

O resultado deve ser gravado num ficheiro JSON.

Exemplo de output esperado:

```
{
   "Ciclismo": { "dificuldade": "Fácil", "horario":["08:00 - 19:00","09:00 - 12:00"], instrutor:"João S
   "Natação": { "dificuldade": "Médio", "horario":["14:00 - 18:00","09:00 - 12:00"], instrutor:"Bruna O
    (...)
}
```

3 Spacy

Implemente um programa "pbempb"[1] que, dado um ficheiro de texto, e dado um padrão,

- procura todas as frases que contêm esse padrão,
- extrai os lemmas dos verbos nela contidos, e extrai as entidades,
- calculando a frequência;
- elimine os verbos auxiliares (suponha existente um conjunto de stopthings que os contenha)
- por fim mostra (stdout) o top 20 de Entidades e verbos.

Exemplo:

```
$ pbempb "JOIN|Jornadas" noticiasJN noticiasDM

Braga, Paulo Novais, U. Minho, Eurotux
decorrer, dizer, increver, organizar, intrevistar, contratar
```

4 Smart grep de nomes próprios

Pretende-se construir um modo versátil de procurar nomes próprios. Nomeadamente:

- o último apelido deve estar sempre completo,
- o primeiro nome deve estar presente, por inteiro ou abreviado
- os nomes intermédios são opcionais, podendo estar ausentes, presentes ou abreviados.

Ao procurar: "José João Dias Almeida" é compatível com (J. Almeida), (J. João Almeida) e de (J. J. Almeida) mas não com (J. F. Almeida)

1. Escreva manualmente uma expressão regular que encontre as ocorrências de "José João Dias Almeida" de acordo com os critérios enunciados
2. Construa um programa que dado um texto, o anote acrescentando parentesis em volta das ocorrências compatíveis encontradas.
3. Escreva uma função Python que dado um nome próprio, calcule a expressão regular a usar de acordo com os critérios descritos.

5 nlgrep

Pretende-se construir um filtro nlgrep (natural language aware grep) que dado um padrão (... de que tipo?)e um texto, procure as respectivas (.... palavras, frases, parágrafos, ?)

1. Sem implementar nada indique:

- Que tipo de padrões nlgrep poderia fazer sentido?
- Que opções escolheria para o nlgrep?
- Que abordagem geral faria para a implementação, que módulos usaria?
- Apresente um conjunto de exemplos de uso.

2. Apresente um algoritmo em pseudo Python, do nlgrep

[1] pbempb: para bom entendedor meia palavra basta"

6 Filtros Unix

Suponha que tem um livros em que os capítulos estão marcados com "== Capítulo ...". Pretende-se extrair os anos por capítulo. Escreva uma sequência de comandos capazes de mostra algo semelhante a:

```
$ ... Amor_de_perdição.txt ...
  == Capítulo I : 1823 1818
  == Capítulo II :
  == Capítulo III : 1830 1830 1832
  ...
  == Capítulo XIV : 1862 1861 1864
```