

Projeto 2 - RI

João Vasconcelos(jvsn)
Rodrigo Carneiro(rcrs3)

Arquivo Invertido

Utilizamos dois scripts:

- Um script utiliza o extrator para conseguir os atributos de cada página e salva um JSON no formato “atributo”: valor
- Outro script lê os JSONs e cria arquivos de posting:
 - Um arquivo será no formato termo: (frequência, documento)
 - O outro arquivo será valor.chave: documento

Arquivo Invertido

- Para calcular a frequência de cada palavra:
 - CountVectorize() do sklearn

```
def frequencyDocument(json_dict, file_index):  
    w = []  
    for words in json_dict.values():  
        if words in ['', '---', ' ', '.', ' -']:  
            continue  
        w.append(words)  
  
    matrix = []  
    try:  
        matrix = vectorizer.fit_transform(w).toarray()  
    except:  
        return  
    matrix = matrix.sum(axis = 0)  
  
    for word, index in vectorizer.vocabulary_.items():  
        freqDoc = (int(matrix[index]), int(file_index))  
        if not inv_index_frequency.get(word, False):  
            inv_index_frequency[word] = set()  
            inv_index_frequency[word].add(freqDoc)
```

Arquivo Invertido

— — —

- Para o arquivo com chave-valor também utilizamos contagem por frequência
- <termos>:(frequência,documento)

```
def twoTermsDocument(json_dict, file_index):
    for key, words in json_dict.items():
        if words in ['', '---', ' ', '.', '-']:
            continue
        words = words.strip().split(' ')

        diff_words = set()
        for word in words:
            cont = 0
            if word in ['', '---', ' ', '.', '-']:
                continue
            for word2 in words:
                if(word == word2):
                    cont += 1
            if not word in diff_words:
                freqDoc = (cont, word)
                diff_words.add(freqDoc)
            else:
                continue
        for freq, word in diff_words:
            newKey = word + '.' + key
            freqDoc = (int(freq), int(file_index))
            if not inv_index_twoTermsDocs.get(newKey, False):
                inv_index_twoTermsDocs[newKey] = set()
            inv_index_twoTermsDocs[newKey].add(freqDoc)
```

Índice Invertido

— — —

Frequência

```
'odyssey': {(1, 321), (1, 330), (1, 1000), (1, 2208)},  
'or': {(3, 2083),  
      (2, 1682),  
      (1, 1690),  
      (1, 1357),  
      (4, 751),  
      (3, 1419),
```

Chave-Valor + Frequência:

```
'ball.game': [(1, 7),  
              (1, 8),  
              (1, 9),  
              (1, 10),  
              (1, 11),  
              (1, 12),  
              (1, 13),  
              (1, 14),  
              (1, 15),  
              (1, 17),  
              (1, 20),  
              (1, 2298)],  
'fighterz.game': [(1, 7),  
                  (1, 8),  
                  (1, 9),  
                  (1, 10),  
                  (1, 11),
```

Índice Invertido

— — —

- Tamanho dos arquivos(sem compressão):

```
print('Size of file Frequency: %.2f kB'%(os.path.getsize('frequency.json')/1024.0))  
print('Size of file TwoTerms: %.2f kB'%(os.path.getsize('twoTerms.json')/1024.0))|
```

Size of file Frequency: 975.81 kB

Size of file TwoTerms: 1223.28 kB

- Tamanho dos arquivos(com compressão):

```
print('Size of file Frequency: %.2f kB'%(os.path.getsize('frequencyCompressed.json')/1024.0))  
print('Size of file TwoTerms: %.2f kB'%(os.path.getsize('twoTermsCompressed.json')/1024.0))
```

Size of file Frequency: 806.29 kB

Size of file TwoTerms: 1042.34 kB

Índice Invertido

— — —

- Stemming: Utilizamos stemming para gerar os arquivos de índice invertido onde os termos estejam normalizados
 - Motivo: queries como game x games x gaming

Processamento de Consulta

— — —

- Uma classe `QueryProcessor()` que recebe uma string como query e gera resultados
- TF-IDF opcional. Utiliza uma flag para decidir
- Limpeza da query por acentos, sinais, stopwords
- Query por string composta
 - ex: `'dark souls' != "dark souls"`

Processamento de Consulta

— — —

```
qp.query("windows hard music", useTfidf=True)|
```

```
[(1543, 0.1472699057818264),  
(1392, 0.14137910955055333),  
(1540, 0.14137910955055333),  
(2708, 0.14137910955055333),  
(1560, 0.1359414514909167),  
(1842, 0.1359414514909167),  
(536, 0.0968350065414749),  
(1221, 0.09301257207273246),  
(1882, 0.09301257207273246),  
(1504, 0.09062763432727779),  
(1733, 0.09062763432727779)]
```

```
qp.query("dota")
```

```
[(2399, 0.028846153846153848),  
(1, 0.0),  
(2, 0.0),  
(3, 0.0),  
(4, 0.0),  
(5, 0.0),  
(6, 0.0),  
(7, 0.0),  
(8, 0.0),  
(9, 0.0),  
(10, 0.0)]
```

```
qp.query("dark souls")
```

```
[(774, 0.05128205128205128),  
(637, 0.03896103896103896),  
(771, 0.03571428571428571),  
(2014, 0.03508771929824561),  
(773, 0.034482758620689655),  
(772, 0.03389830508474576),  
(849, 0.030303030303030304),  
(2048, 0.030303030303030304),  
(419, 0.02857142857142857),  
(2193, 0.023255813953488372),  
(1807, 0.02)]
```

Processamento de Consulta

- Comparação entre o uso do TF-IDF utilizando correlação de Spearman

$$S(\mathcal{R}_1, \mathcal{R}_2) = 1 - \frac{6 \times \sum_{j=1}^K (s_{1,j} - s_{2,j})^2}{K \times (K^2 - 1)}$$

Processamento de Consulta

— — —

```
def getSumSquareDist(r1, r2):  
    result = 0  
    docs = list(r1.keys()) + list(r2.keys())  
    for doc in docs:  
        squareDistance = (r1.get(doc, 0) - r2.get(doc, 0)) ** 2  
        result += squareDistance  
    return result
```

```
def spearmanCorrelation(sumSquareDist, k):  
    num = 6 * sumSquareDist  
    den = k * (k ** 2 - 1)  
    return 1 - (num / den)
```

Processamento de Consulta

— — —

- Resultado:

```
Query #("dark souls"):  
Sum Square Distance = 0.10  
0.9999453705575814  
Query #(dota):  
Sum Square Distance = 0.08  
0.9999545669919276  
Query #(hard game):  
Sum Square Distance = 0.13  
0.9999284734695677  
Query #(challenge):  
Sum Square Distance = 0.28  
0.9998423159174965  
Query #(dark windows 2gb ram):  
Sum Square Distance = 0.47  
0.9997352341144584
```

Processamento de Consulta

— — —

- Query utilizando os atributos
 - Classe `GeneralQuery()` recebe além da string geral da query, valores dos atributos para filtrar a busca
- TF-IDF opcional também

Processamento de Consulta

— — —

```
queryTFIDF = GeneralQuery("Creed", '', 'windows', '', '', '', useTfIdf=False)  
queryTFIDF.processQuery()
```

```
[('2506', 0.017361111111111112),  
 ('2712', 0.015151515151515152),  
 ('2594', 0.014285714285714285),  
 ('291', 0.014285714285714285),  
 ('2592', 0.014285714285714285),  
 ('2593', 0.014285714285714285),  
 ('1523', 0.013888888888888888),  
 ('292', 0.013888888888888888),  
 ('1522', 0.013605442176870748),  
 ('2345', 0.011904761904761904),  
 ('2342', 0.011904761904761904),  
 ('880', 0.007575757575757576),  
 ('890', 0.007246376811594203),  
 ('878', 0.007246376811594203),  
 ('877', 0.006944444444444444),  
 ('926', 0.006944444444444444),  
 ('887', 0.006944444444444444),  
 ('884', 0.006666666666666667),  
 ('876', 0.006666666666666667),  
 ('885', 0.006666666666666667),  
 ('886', 0.006666666666666667),  
 ('883', 0.006666666666666667)]
```

```
query = GeneralQuery("Creed", '', 'windows', '', '', '', useTfIdf=True)  
query.processQuery()
```

```
[('880', 0.030318747181659125),  
 ('890', 0.029000540782456553),  
 ('878', 0.029000540782456553),  
 ('877', 0.02779218491652086),  
 ('926', 0.02779218491652086),  
 ('887', 0.02779218491652086),  
 ('884', 0.026680497519860027),  
 ('876', 0.026680497519860027),  
 ('885', 0.026680497519860027),  
 ('886', 0.026680497519860027),  
 ('883', 0.026680497519860027),  
 ('2506', 0.006727865696734669),  
 ('2712', 0.00587159188078662),  
 ('2594', 0.00553607234474167),  
 ('291', 0.00553607234474167),  
 ('2592', 0.00553607234474167),  
 ('2593', 0.00553607234474167),  
 ('1523', 0.005382292557387734),  
 ('292', 0.005382292557387734),  
 ('1522', 0.005272449852134924),  
 ('2345', 0.004613393620618058),  
 ('2342', 0.004613393620618058)]
```

Processamento de Consulta

— — —

- Resultado:
 - Query("Creed", "windows"): 0.9999992112358141
 - Query("Sims", "windows"): 0.9999998532640239
 - Query("Crash", "2 gb"): 0.9999996114138657
 - Query("Dark Souls", "4 gb"): 0.9999993118481989
 - Query("Player unknown", "windows"): 0.9999995890471487

Interface

- Tkinter python

Engenho de Busca - Jogos

Nome do Jogo
dark souls

Description

Operational System

RAM

Storage

Price

Pesquisar

Resultados:

```
Name: dark souls_ç_iii
Price: 39,97
os: windows 7 spl 64bit windows 81 64bit windows 10 64bit
ram: 4 gb ram
storage: 25 gb available space
URL: https://store.steampowered.com/app/374320/?snr=1_5_9__205
```

```
Name: dark souls_ç_iii - the ringed city_ç
Price: 59,90
os: windows 7 spl 64bit windows 81 64bit windows 10 64bit
ram: --
storage: 25 gb available space
URL: https://store.steampowered.com/app/374320/?snr=1_5_9__205
```

```
Name: tabletop simulator - battle for souls
Price: 5,24
```