Perfil EL - Engenharia de Linguagens
(1º ano do MEI)
Scripting e Processamento de Linguagem Natural

Trabalho Prático 2

Relatório de Desenvolvimento

João Gonçalves
PG46535

Sara Queirós
PG47661

22 de junho de 2022

Conteúdo

Capítulo 1

O que é Sentiment Analysis?

Sentiment Analysis é uma técnica que permite analisar um pedaço de texto para determinar o sentimento por trás do mesmo. Por norma, combina machine learning e processamento de linguagem natural (NLP) para o conseguir. Através desta técnica, um programa consegue determinar se o sentimento associado a um pedaço de texto é positivo, negativo ou neutro. É uma técnica poderosa que tem aplicações empresariais importantes. como por exemplo, analisar o feedback de clientes. Depois de recolher esse feedback através de vários meios como o Twitter e o Facebook, uma empresa pode empregar algoritmos de Sentiment Analysis nesses excertos de texto para entender a atitude dos seus clientes em relação ao seu produto ou serviço.

Capítulo 2

A nossa abordagem: Sistema Baseado em Regras

Ao contrário dos modelos automatizados, as abordagens baseadas em regras dependem de regras personalizadas para classificar os dados. Técnicas populares para este efeito incluem tokenização, parsing, cauling, entre outras. Uma vantagem dos sistemas baseados em regras é a capacidade de personalizção, podendo estes ser baseados em contextos mais específicos, desenvolvendo regras mais inteligentes.

2.1 Projeto

O nosso projeto consiste no emprego de técnicas de sentiment analysis baseada em regras para analisar os títulos de notícias retiradas do site https://www.publico.pt/, procedendo depois ao tratamento dos dados resultantes dessa análise. Desta forma, é-nos possível tirar conclusões quanto à variação dos sentimentos dentro das próprias notícias, comparar sentimentos entre as várias notícias, para além de tecer padrões ao longo do tempo das mudanças nos sentimentos.

2.1.1 Tratamento do Dataset

O nosso projeto consiste no emprego de técnicas de sentiment analysis baseada em regras para analisar os títulos de notícias retiradas do site https://www.publico.pt/, procedendo depois ao tratamento dos dados resultantes dessa análise. Desta forma, é-nos possível tirar conclusões quanto à variação dos sentimentos dentro das próprias notícias, comparar sentimentos entre as várias notícias, para além de tecer padrões ao longo do tempo das mudanças nos sentimentos.

2.1.2 Estratégia de Análise de Sentimentos

Começamos por definir os 3 principais tipos de sentimentos: bom, mau e neutro. A partir daí, aproveitamos o facto de que os sentimentos no dataset SentiLex estão cotados numa escala de [-1, 1] e utilizamos esta mesma escala nas nossas análises.
De seguida, ciramos um dataset adicional com palavras que classificamos como 'MULTIPLICADO-

RAS' (ex.: muito, pouco, ...), visto que estas palavras aumentam ou diminuem o valor do sentimento da palavra que as procede.

```
Fórmula: [Valor da Palavra Multiplicadora] x [Sentimento da Palavra]
```

Como existem expressões cujas palavras individualmente possuem um sentimento diferente de quando estão combinadas numa mesma frase. Por isso, dividimos o dataset de modo a ter um dataset apenas para expressões e outro com todas as palavras individuais do SentiLex. De modo a proceder a esta divisão, foram empregados os conhecimentos adquiridos na UC de SPLN, para desenvolver um script que faça essa divisão, visto que manualmente implicaria percorrer todas as entradas do SentiLex, procurar as expressões removê-las e colocá-las noutro dataset.

```
import csv
import re
import pandas as pd

#   Funo   que procura todas alinhas do SentiLex.csv com espaos [\s]+ e as coloca numa
    lista de strings

def LerParaLista() :

    with open('SentiLex.csv', "r", encoding='utf-8') as file:
        data = file.read().splitlines()
        lista = []
        for row in data :
            if (re.search("[\s]+", row)) :
                lista.append(row)
        return (lista)

# Gera os csv de Expresses e Palavras

def GerarDocumentos(lista) :
    txt = open('exp.txt','w', encoding='utf-8')
    for elem in lista :
        txt.write(elem + '\n')
    txt.close()

    #Transformar num csv
    dataframe1 = pd.read_csv("exp.txt", delimiter=',')
    dataframe1.to_csv('Express.csv', index = None, header=['word', 'sent'])

    with open('SentiLex.csv', "r", encoding='utf-8') as inp :
        txt = open('tratado.txt','w', encoding='utf-8')
        for row in csv.reader(inp):
            #Tratamento das linhas para ficar igual s do .csv
            row = str(row)
            row = re.sub(r'\[|\]', '', row)
```

```
            row = re.sub(r'\'', '', row)
            row = re.sub(r', ', ',', row)
            flag = 0
            # Se a expresso estiver na lista, no escreve no novo ficheiro, pois flga
                fica a 1

            for exp in lista :
                if (exp == row) :
                    flag = 1

            if(flag == 0) :
                #print('Escrever no ficheiro: ' + row)
                txt.write(row + '\n')
        txt.close()
    #Transformar num csv
    dataframe1 = pd.read_csv("tratado.txt", delimiter=',')
    dataframe1.to_csv('SentiLex2.csv', index = None)

GerarDocumentos(LerParaLista())
```

Posto isto, e havendo os datasets necessários (SentiLex.csv, exp.csv e mult.csv) para processar os sentimentos associados às palavras em causa, começamos por, efetivamente, desenvolver a solução, seguindo 3 passos:

- Processamento de Informação
- Cálculo do Sentimento
- Apresentação de resultados gráficos

2.2 Tratamento da Informação

Uma vez que o alvo de análise do trabalho trabalho é o site de notícias do jornal 'Público', foi necessário efetuar análise e processamento da página com o intuito de extrair apenas os títulos das notícias. Para tal, foi utilizada a biblioteca **BeatifulSoup**:

```
def treatInfo(url):
    final = []
    response = requests.get(url)

    soup = BeautifulSoup(response.text, 'html.parser')
    #Retirar todas as headlines do site para analisar os sentimentos
    headlines = soup.find('body').find_all('h4', 'headline')
    for x in headlines:
        x = x.text.strip()
        final.append(x)
    return final
```

```
url='https://www.publico.pt/online'

print(treatInfo(url))
```

Este procedimento foi realizado de forma externa ao programa, com o intuito de permitir a padrozinação do input que ele irá receber. Deste modo, esta função escreve num ficheiro uma lista de strings, com as palavras/frases/textos que se pretende analisar, sendo, por isso, o modelo que se deve seguir para utilizar o programa.

2.3 Cálculo de Sentimento

De forma a calcular o sentimento associado a cada título de notícia é necessário seguir os procedimentos:

```
def analyseSents(file):
    content = open(file, 'r').read()
    blocks = tolistString(content)
    for x in blocks:
        general[x] = {'bom': [], 'mau': [], 'neutro': [], 'total': '', 'valor': 0.0}
        x_lem = lemmatization(x)
        x1 = catch_expressions(x, x_lem)
        x2 = catch_mult(x, x1)
        x_WS = remove_mystopwords(x2)
        x3 = catch_words(x, x_WS)
        #similarityWords(x, x3)
    return blocks
```

De forma mais detalhada:

- Inicialmente o ficheiro input é lido e convertido para o formato necessário (lista de strings).
- Para cada uma das frases a ser analisada, é criado uma entrada no dicionário, onde será registada a lista de palavras/expressões com sentimentos, bons, maus e neutros e, futuramente, o valor final da análise efetuada.
- De forma a obtermos facilitar o processamento do sentimento de cada frase (não havendo preocupações com conjugação de verbos, género e sigular/plurar), estas foram sujeitas ao processo de *lemmatization*.
- Feito isto, foi iniciado o processo efetivo de cálculo de sentimentos: procurar expressões na frase que possuam sentimento. Após obtidas e removidas da frase, são procurados multiplicadores, tal como já foi mencionado anteriormente, e as respetivas palavras. Finalmente, dadas as restantes palavras, são removidas as *stop words*(utilizando a biblioteca NLTK) para evitar buscas desnecessárias por sentimentos em palavras comuns da Língua Portuguesa.

- Para além disso, foi elaborada a função "similarityWords"que, para as restantes palavras, calcula a miníma distância entre a palavras em questão e o sentimento "bom"ou "mau". No entanto, os resultados obtidos ao considerar estes valores, afastam-se da realidade e, por isso, foram desconsiderados.

Posteriormente, o sentimento é processado e calculado, seguindo a fórmula:

```
Total = (Soma valores 'BOM + Soma valores 'MAU' ) / Nºelems(Bom + Mau + Neutro)
```

Este valor é guardado no dicionário mencionado inicialmente, juntamente com a label correspondente "POSITIVO", "NEGATIVO"ou "NEUTRO".
Por fim, é efetuada e apresentada uma comparação entre os resultados obtidos pelo programa e os esperados, considerando a opinião geral do grupo, face aos títulos das notícias. Estes resultados são guardados num ficheiro JSON, para poderem ser processados por outro programa.

2.4 Apresentação de resultados gráficos

Dado o ficheiro JSON que é escrito no final da execução do programa anterior, este é interpretado e armazenado num dicionário. Com estas informações, é desenhado um gráfico que relaciona no eixo y, o valor do sentimento da notícia, e no eixo x, a posição da notícia ao longo da página.

Capítulo 3

Testes

3.1 Como correr os programas?

Para realizar a extração e preparação da informação do link em causa, basta:

```
python prepInfo.py >> resultado.txt
```

Para efetuar a análise de sentimento desse resultado:

```
python SA.py resultado.txt
#Caso o utilizador pretenda visualizar o resultado das comparacoes entre os
   resultados
#obtidos e esperados, quando oportuno devera selecionar a opcao 1 e mencionar
#que o ficheiro em causa e:
comparacao.txt
```

Dados os resultados, é possível elaborar um gráfico que permita visualizar perceber os valores obtidos.

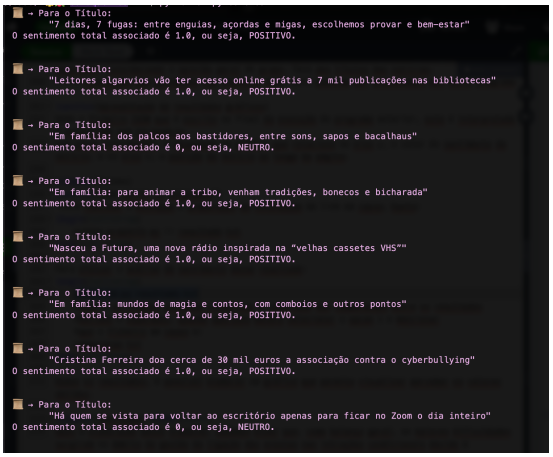

Figura 3.1: Execução do Programa de Análise de Sentimentos

Figura 3.2: Comparação com os resultados esperados

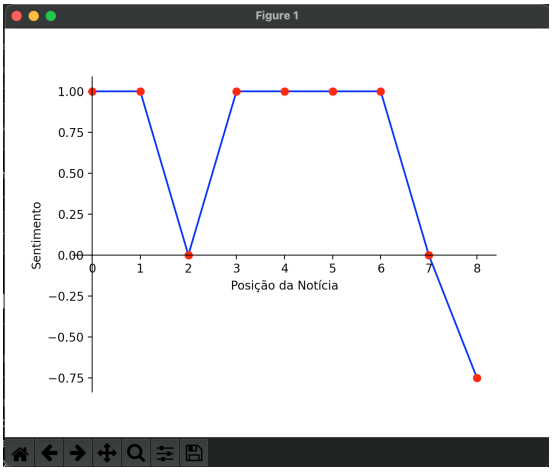

Figura 3.3: Gráfico relativo aos sentimentos

Capítulo 4

Conclusão

Após a elaboração deste trabalho, podemos dizer, como balanço geral, que grande parte do conhecimento obtido ao longo desta UC foi aplicado, ajudando-nos a ter a perceção da abrangência deste tipo de bibliotecas, e até mesmo, a forma como é possível interligar conceito e, assim, expandir o trabalho elaborado.
Para além disso, o tema *Sentiment Analysis* , forçou-nos a expandir as nossas perspetivas face ao significado das palavras em função do contexto em que se inserem.