

# Trabalho Prático 2 - *emojis*

Simão Monteiro<sup>1</sup>[PG47677] and Nuno Silva<sup>2</sup>[PG42645]

Universidade do Minho, Gualtar, Portugal

## 1 Introdução

Este projeto foi realizado no âmbito da unidade curricular de Scripting no Processamento de Linguagem Natural, de forma a consolidar a matéria lecionada nas aulas. Foi dado a escolher aos alunos vários temas, sendo que o grupo escolheu o tema correspondente aos *emojis*. Para isso devia ser desenvolvido um programa na linguagem *Python*, com módulos prontos para manipular *emojis*. De forma a alimentar o nosso programa, usamos o ficheiro csv de *tweets* fornecido ao longo do ano letivo pelos docentes.

## 2 Funcionalidades

### 2.1 Extração e Identificação dos *emojis* do ficheiro CSV

Para extrair *emojis* do ficheiro *tweets* criou-se uma função chamada ***extract\_emojis***, nesta função é devolvido o *emoji*, separado do *tweet* original. Para identificar o *emoji* usamos uma biblioteca chamada *emot.emo unicode*, se alguma elemento do *tweet* estiver nessa biblioteca, então esse elemento é um *emoji*.

```
def extract_emojis(s):  
    return ''.join(c for c in s if c in emoji.UNICODE_EMOJI['pt'])
```

Fig. 1. Função *extract\_emojis*

### 2.2 Extração e Identificação dos *emoticons* do ficheiro CSV

Para identificar os *emoticons* nos *tweets*, criamos uma função chamada ***find\_emoticons***, onde usamos uma biblioteca de *emoticons*, *Emoticon Dict.p*, que consegue identificar se o elemento do texto é ou não um *emoticon*, caso seja devolve esse *emoticon*.

```
def find_emoticons(text):
    emoticon_pattern = re.findall(u'(' + u'|'.join(k for k in Emoticon_Dict) + u')', text)
    aux=[]
    for elem in emoticon_pattern:
        elem = elem.replace('-', '')
        if elem != '':
            aux.append(str(elem))
    return aux
```

Fig. 2. Função *find emoticons*

### 2.3 Converter *emojis* em texto e texto em *emojis*

Nesta função usamos o módulo *emoji*, que contém uma função chamada *emojize* e *demojize*. Na função ***convert text to emojis***, recebemos um texto que tem de estar entre dois pontos e ter um certo formato *standarizado*. Depois usando a função *emojize* juntamente com a *flag* *pt*, para avisar que queremos usar a Língua Portuguesa, devolve-nos o *emoji*.

```
def convert_text_to_emojis(text):
    return emoji.emojize(text, language='pt')
```

Fig. 3. Função *Convert text to emojis*

No caso da função ***convert emojis to text***, é semelhante à anterior só que em vez de usar a função *emojize* usa-se a função *demojize*.

```
def convert_emojis_to_text(text):
    return emoji.demojize(text, language='pt')
```

Fig. 4. Função *Convert emojis to text*

### 2.4 Converter *emoticons* em *emojis*

Na função ***convert emoticons to emojis*** convertemos os *emoticons* em *emojis*. Para isso, e como não foram encontrados módulos *python* capazes de fazer essa conversão, tivemos de fazer uma pequena análise dos *tweets* para ver quais os *emoticons* mais frequentes. Após essa análise foi criado um dicionário onde as chaves eram os *emoticons* e os valores os seus *emojis* correspondentes, este

dicionário tem o nome de ***rawEmojis***. A correspondência dos *emoticons* foi feita através da página da *wikipedia* que nos fornece a versão *emoji* de um *emoticon*, pode ser vista nas referências.

```
rawEmojis = {
    ":)" : "😄",
    "=)" : "😄",
    ":(" : "😞",
    ":D" : "😄",
    ":/ " : "😞",
    ":'( " : "😞",
    ":P" : "😜",
    "XD" : "😜",
    ":3" : "😺",
    "DX" : "😞",
    "d:" : "😞",
    "XP" : "😜",
    "D8" : "👩",
    ":o" : "😞"
}
```

**Fig. 5.** Dicionário para usar na conversão.

Finalmente, dentro da função associamos os *emoticons* aos *emojis* e devolvemos a conversão.

```
def convert_emoticons_to_emoji(text):
    words = text.split(" ")
    outcome = " "
    for word in words:
        outcome += rawEmojis.get(word, word) + " "
    return(outcome)
```

**Fig. 6.** Função para converter *emoticons* em *emojis*

## 2.5 Converter *emoticons* em texto

Nesta função também usamos a biblioteca *emoji* com a função *demojize*, só que antes de procedermos ao *demojize* do texto, convertamos à conversão dos *emoticons* através da função já descrita (***convert emoticons to emoji***).

```
def convert_emoticons_to_text(text):
    return emoji.demojize(convert_emoticons_to_emoji(text))
```

**Fig. 7.** Função para converter *emoticons* em texto.

## 2.6 Procurar Taxonomia

De forma a obtermos a taxonomia de cada *emoji* usamos a biblioteca *advertools*, que tem muitas funcionalidades, uma delas é obter a taxonomia dos *emojis*. A função *emoji search* permite-nos, entre outras coisas, obter o grupo associado ao *emoji* em questão.

```
def find_taxonomia(text):
    result=dict()
    aux=extract_emojis(text)
    for emoji in aux:
        taxonomia[emoji]= adv.emoji_search(emoji)['group'][0]
    return result
```

**Fig. 8.** Função para procurar a taxonomia de um certo *emoji*.

## 2.7 Embedding com *emojis*

Para realizar este ponto baseamo-nos no que nos foi mostrado nas aulas sobre *embeddings*. Inicialmente foi necessário passar o conteúdo dos tweets, originalmente em csv, para um ficheiro .txt pois o treino do modelo não estava a aceitar um csv como input. Após passar o conteúdo para o ficheiro .txt foi também necessário transformar os *emojis* em texto, e os *emoticons* (*emojis* textuais) também em texto, transformando-os primeiro em *emojis* e de seguida passando-os de *emojis* para texto através da função *demojize* do módulo *emojis*. Após treinado o modelo, para que o output fossem *emojis* e não as suas versões textuais, voltou a fazer-se a conversão de texto para emoji.

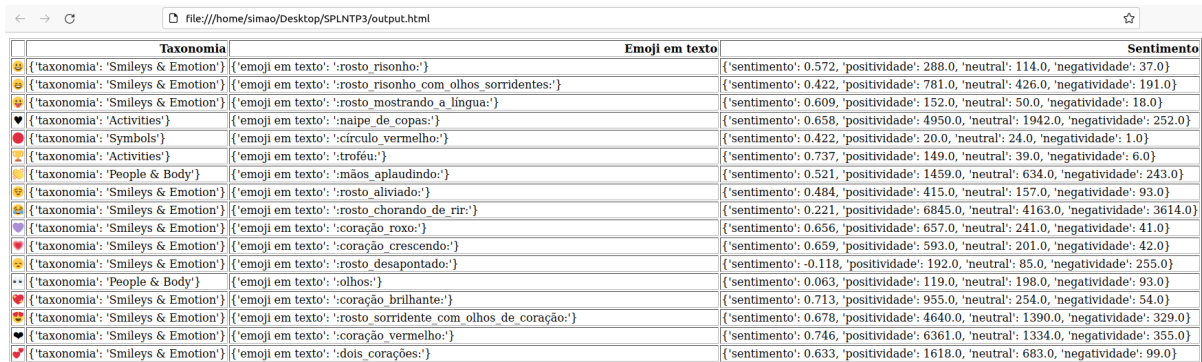
Ficamos satisfeitos com os resultados do embedding para alguns *emojis* sendo que para outros, o output não faz qualquer sentido. Inicialmente pensamos que elementos como identificações de utilizadores no tweet, links url e outro tipo de elementos sem significado semântico pudessem estar a estragar o output no entanto após a remoção de alguns desses elementos os resultados até se demonstraram piores para *emojis* que antes tinham resultados satisfatórios. Dado este insucesso, pensamos que para alguns *emojis* os resultados não são satisfatórios porque de uma forma geral devem ser pouco utilizados pelos utilizadores nos seus tweets e então ficam "pouco treinados" pelo modelo.

```
def semelhantes(emo):
    aux=[]
    a = emoji.demojize(emo, language='pt')
    b = re.sub(':', '', a)
    res=model.wv.most_similar(b)
    for (a,c) in res:
        a = ":" + a + ":"
        a=emoji.emojize(a,language='pt')
        a=re.sub(':', '', a)
        aux.append((a,c))
    return aux
```

Fig. 9. Função para semelhantes para *emoji embedding*

### 3 Resultado Final

Existem no fundo dois *scripts* um para gerar um ficheiro *html* que consiste numa única tabela com toda a análise feita por nós.



	Taxonomia	Emoji em texto	Sentimento
😊	{'taxonomia': 'Smileys & Emotion'}	{'emoji em texto': 'rosto risonho:'}	{'sentimento': 0.572, 'positividade': 288.0, 'neutral': 114.0, 'negatividade': 37.0}
😄	{'taxonomia': 'Smileys & Emotion'}	{'emoji em texto': 'rosto risonho com olhos sorridentes:'}	{'sentimento': 0.422, 'positividade': 781.0, 'neutral': 426.0, 'negatividade': 191.0}
😛	{'taxonomia': 'Smileys & Emotion'}	{'emoji em texto': 'rosto mostrando a língua:'}	{'sentimento': 0.609, 'positividade': 152.0, 'neutral': 50.0, 'negatividade': 18.0}
♥	{'taxonomia': 'Activities'}	{'emoji em texto': 'naipe de copas:'}	{'sentimento': 0.658, 'positividade': 4950.0, 'neutral': 1942.0, 'negatividade': 252.0}
🔴	{'taxonomia': 'Symbols'}	{'emoji em texto': 'círculo vermelho:'}	{'sentimento': 0.422, 'positividade': 20.0, 'neutral': 24.0, 'negatividade': 1.0}
🏆	{'taxonomia': 'Activities'}	{'emoji em texto': 'troféu:'}	{'sentimento': 0.737, 'positividade': 149.0, 'neutral': 39.0, 'negatividade': 6.0}
👏	{'taxonomia': 'People & Body'}	{'emoji em texto': 'mãos aplaudindo:'}	{'sentimento': 0.521, 'positividade': 1459.0, 'neutral': 634.0, 'negatividade': 243.0}
😌	{'taxonomia': 'Smileys & Emotion'}	{'emoji em texto': 'rosto aliviado:'}	{'sentimento': 0.484, 'positividade': 415.0, 'neutral': 157.0, 'negatividade': 93.0}
😭	{'taxonomia': 'Smileys & Emotion'}	{'emoji em texto': 'rosto chorando de rir:'}	{'sentimento': 0.221, 'positividade': 6845.0, 'neutral': 4163.0, 'negatividade': 3614.0}
💜	{'taxonomia': 'Smileys & Emotion'}	{'emoji em texto': 'coração roxo:'}	{'sentimento': 0.656, 'positividade': 657.0, 'neutral': 241.0, 'negatividade': 41.0}
💖	{'taxonomia': 'Smileys & Emotion'}	{'emoji em texto': 'coração crescendo:'}	{'sentimento': 0.659, 'positividade': 593.0, 'neutral': 201.0, 'negatividade': 42.0}
😞	{'taxonomia': 'Smileys & Emotion'}	{'emoji em texto': 'rosto desapontado:'}	{'sentimento': -0.118, 'positividade': 192.0, 'neutral': 85.0, 'negatividade': 255.0}
👁	{'taxonomia': 'People & Body'}	{'emoji em texto': 'olhos:'}	{'sentimento': 0.063, 'positividade': 119.0, 'neutral': 198.0, 'negatividade': 93.0}
💎	{'taxonomia': 'Smileys & Emotion'}	{'emoji em texto': 'coração brilhante:'}	{'sentimento': 0.713, 'positividade': 955.0, 'neutral': 254.0, 'negatividade': 54.0}
😍	{'taxonomia': 'Smileys & Emotion'}	{'emoji em texto': 'rosto sorridente com olhos de coração:'}	{'sentimento': 0.678, 'positividade': 4640.0, 'neutral': 1390.0, 'negatividade': 329.0}
❤	{'taxonomia': 'Smileys & Emotion'}	{'emoji em texto': 'coração vermelho:'}	{'sentimento': 0.746, 'positividade': 6361.0, 'neutral': 1334.0, 'negatividade': 355.0}
❤❤	{'taxonomia': 'Smileys & Emotion'}	{'emoji em texto': 'dois corações:'}	{'sentimento': 0.633, 'positividade': 1618.0, 'neutral': 683.0, 'negatividade': 99.0}

Fig. 10. *Output* com análise dos *emojis*.

Cada coluna tem o seguinte significado:

- A coluna 0 corresponde à taxonomia do *emoji* em questão.
- A coluna 1 corresponde ao *emoji*, em texto escrito (traduzido para português).
- A coluna 2 corresponde à análise da polaridade e sentimento do *emoji* em questão.

### 3.1 Polaridade e Sentimento do *Emoji*

Em relação á polaridade , este módulo *emosent*, baseia-se no projeto *emoji-sentiment* escrito na linguagem *javascript*. Neste caso em relação a cada valor tem o seguinte significado:

- Negatividade corresponde ao número absoluto do número de ocorrências do *emoji* de *tweets* rotulados como negativos.
- Neutral corresponde ao número absoluto do número de ocorrências do *emoji* de *tweets* rotulados como neutros.
- Positividade corresponde ao número absoluto do número de ocorrências do *emoji* de *tweets* rotulados como positivos.

É expectável que quanto maior o valor, maior a probabilidade de o *tweet* ter a respetiva polaridade. Em relação ao valor atribuído ao sentimento, foi baseado num estudo feito por Kralj Novak, Petra; Smailović, Jasmina; Sluban, Borut and Mozetič, Igor em 2015 onde pediram a 83 humanos que rotulassem *tweets* com *emojis* como positivos, neutros e negativos. Isto fez gerar um valor compreendido de (-1 a 1) que corresponde a quão positivo, a quão negativo e a quão neutro é o *emoji* *correspondente*.

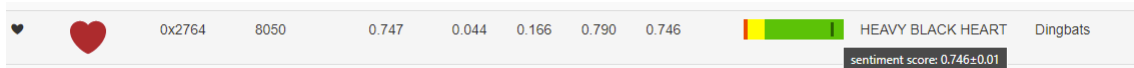


Fig. 11. Pontuação do sentimento, neste caso positivo, no *emoji* em questão.



Fig. 12. Pontuação do sentimento, neste caso negativo, no *emoji* em questão.

### 3.2 Embedding com *emojis*

Exemplo de output de embedding para o emoji do coração vermelho.

```
67 print(semelhantes('❤️'))
```

```
(base) simao@simao-OMEN-by-HP-Laptop:~/Desktop/SPLNTP3$ python embedding.py
[('❤️', 0.7644709348678589), ('♥️', 0.7620498538017273), ('💖', 0.7021562457084656), ('💜', 0.6614087224006653), ('😞', 0.6262532472610474), ('😓', 0.6239866614341736), ('😬', 0.6212470531463623), ('💕', 0.6204748749732971), ('💎', 0.615935742855072), ('💩', 0.6068136096000671)]
```

Fig. 13. Embedding para um coração vermelho

## 4 Conclusão

A realização deste trabalho ajudou-nos a conhecer melhor alguns módulos python que ainda não conhecíamos muito bem. Para além disso foi também interessante explorar as diferentes funcionalidades dos módulos e o que é possível criar com elas. Foi um trabalho interessante de realizar dado que deu para misturar estes módulos com alguns conceitos lecionados nas aulas.

## 5 GitHub

<https://github.com/nuno1197/SPLNTP3.git>

## References

1. <https://github.com/dematerializer/emoji-sentiment>
2. <https://www.clarin.si/repository/xmlui/handle/11356/1048>
3. <https://medium.com/geekculture/text-preprocessing-how-to-handle-emoji-emoticon-641bbfa6e9e7>
4. <https://www.section.io/engineering-education/how-to-convert-text-to-emoji-using-python/>
5. [https://en.wikipedia.org/wiki/List\\_of\\_emoticons](https://en.wikipedia.org/wiki/List_of_emoticons)
6. <https://advertools.readthedocs.io/en/master/advertools.emoji.html>
7. Projeto de word-embeddings com 5,5Gb de tweets de um utilizador:  
<https://www.youtube.com/watch?v=5PL0TmQhItYt=717s>
8. <https://embeddings.machheads101.com/>