Scripting no Processamento
de Linguagem Natural (1º ano de MEI)

Trabalho Prático

Relatório de Desenvolvimento

Ana Sofia Gomes
(PG47003)

Pedro Pereira
(A80376)

22 de junho de 2022

Conteúdo

Capítulo 1

Introdução

No âmbito da unidade curricular Scripting no Processamento de Linguagem natural, os docentes desafiaram os alunos a escolher uma das temáticas apresentadas e a desenvolver uma aplicação com base na mesma.

Assim, a equipa optou por selecionar a temática *sentiment analysis* com o intuito de construir uma aplicação capaz de oferecer um conjunto de estatísticas sobre um dado *post* de Instagram.

Pretendia-se que esta tivesse principal foco na análise de sentimentos dos comentários, de forma a complementar as funcionalidades já oferecidas pelo Instagram.

Estrutura do Relatório

O presente relatório explicita detalhadamente o processo de desenvolvimento do projeto e a abordagem escolhida para solucionar o problema em mãos.

Inicialmente, em 2.1, são discutidos os dados utilizados na aplicação, bem como a forma como foram preparados.

Depois disto, é descrita a forma como são calculados os sentimentos de cada frase a partir dos dados previamente referidos.

Em 2.2.2, são expostos os resultados do treino do modelo criado a partir do *dataset* de *tweets* utilizado nas aulas.

Na secção 2.3, é feito um resumo da metodologia utilizada para obter dados relativos a uma publicação do *Instagram*.

Por fim, colmata-se o documento com a apresentação da aplicação final e um exemplo de uma página HTML criada pela mesma.

Capítulo 2

Conceção/Desenho da Resolução

2.1 Preparação dos dados

Antes de se proceder à construção do analisador de sentimentos, tornou-se necessário recolher e tratar um conjunto de informações que iriam servir como referência durante o processo de análise.

2.1.1 Dicionários de palavras

Como base para a análise, foi utilizado um *dataset* que contém uma extensa lista de palavras positivas e negativas. Embora este *dataset* tenha sido criado no contexto financeiro e esteja em inglês, este pode ser utilizado neste projeto, desde que seja propriamente processado.

Para isso, foi primeiro dividido o ficheiro original em dois: um com apenas as palavras positivas e outro com as palavras negativas. Posteriormente, foram removidos os caracteres extra, como é o caso dos espaços.

Depois deste pré-processamento, foram traduzidas todas as palavras para português recorrendo à API do tradutor da Google. Tendo já todas as palavras em português, procedeu-se à criação um dicionário em Python em que cada par ($chave : valor$) é constituído por uma palavras e o sentimento que lhe é atribuído.

Como não se possui informação sobre quão positiva ou negativa cada palavra é, foi atribuído um valor de 0.5 a todas as palavras, variando o sinal consoante o ficheiro em que estas estavam.

2.1.2 Dicionário de *emojis*

Tendo em consideração que o analisador de sentimentos tinha como principal finalidade avaliar comentários retirados do Instagram, era essencial que este fosse capaz de reconhecer *emojis* e de lhes atribuir um sentimento.

Assim sendo, optou-se por criar um dicionário que possui como chave o *unicode* do *emoji* e como valor o sentimento associado ao mesmo. Para o preencher, foi utilizado um *dataset* baseado num *ranking* de sentimentos de *emojis*, construído através da análise de *tweets*.

Este *dataset* trata-se de um ficheiro CSV em que cada linha possui informações relativas a um dado *emoji*, nomeadamente o número de ocorrências total e o número de ocorrências positivas, neutras e negativas. Assim, para determinar o sentimento de cada *emoji*, basta efetuar o seguinte cálculo:

$$sentimento = \frac{p}{t} - \frac{n}{t}$$

onde:

p = número de ocorrências positivas
n = número de ocorrências negativas
t = número de ocorrências total

2.2 Analisador de sentimentos

Concluída a preparação dos dados, passou-se para a edificação do analisador de sentimentos.

2.2.1 1.ª abordagem:

Numa primeira fase, começou-se por elaborar um analisador exclusivamente baseado em regras. Para cada comentário inserido, este:

- Efetua um tratamento ao comentário, removendo todas as *mentions*[1], *hashtags* e hiperligações;
- Divide o comentário em orações, utilizando como separadores os sinais de pontuação (,;?!.);
- Para cada oração:
  - Cria uma lista vazia para armazenar os multiplicadores, as palavras e os *emojis* presentes na oração que constam nos dicionários criados anteriormente;
  - Divide a oração em palavras e/ou expressões;
  - Para cada expressão/palavra, verifica se esta se encontra presente no dicionário de palavras, multiplicadores ou *emojis*. Em caso afirmativo, esta é adicionada à lista correspondente. Caso contrário, é ignorada. É de notar que, quando uma expressão não pertence a nenhum dos dicionários, esta é fragmentada em palavras. Verifica-se posteriormente se cada uma destas consta em algum dos dicionários;

[1] caracter '@' seguido do *username de um utilizador*

– Calcula o sentimento da oração, primeiro fazendo a média dos sentimentos das palavras encontradas, depois multiplicando todos os valores dos multiplicadores encontrados. Estes dois valores são assimilados segundo a seguinte função:

$$f(s,m) = \begin{cases} s * m & \text{se } |m| \leq 1 \\ s + (1 - |s|) * \frac{1-m}{m} * \text{sign}(s * m) & \text{se } |m| > 1 \end{cases}$$

Em que s é a média dos sentimentos, m o resultado da multiplicação dos multiplicadores, e sign uma função que devolve o sinal do argumento.

Embora complicado, o caso em que $m > 1$ pode ser descrito por três propriedades:

* O resultado está sempre contido no intervalo $[-1, 1]$
* Valores absolutos de m maiores dão origem a resultados maiores
* O sinal de $s * m$ é preservado

- Determina o sentimento do comentário, calculando a média dos sentimentos das orações que compõem o comentário.

A divisão das orações em expressões e palavras é efetuada através do autómato apresentado na figura abaixo.

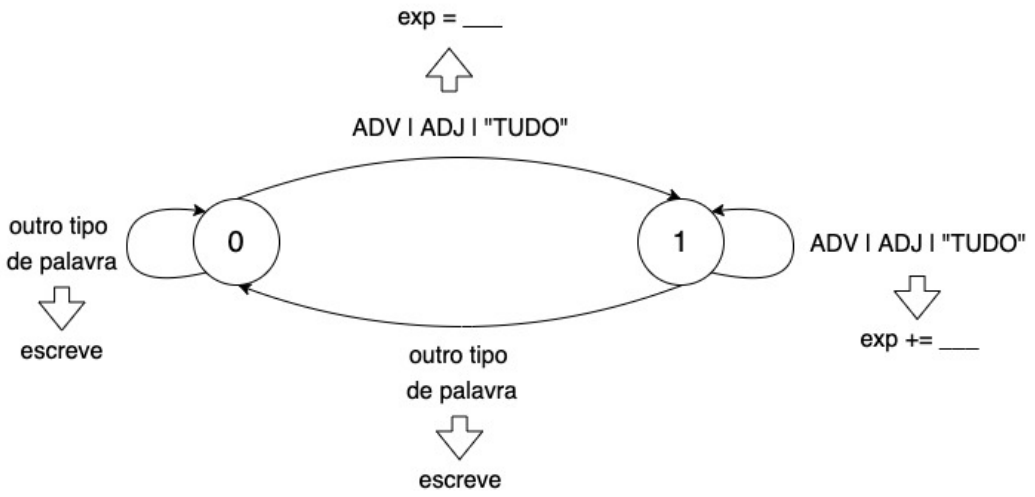

Figura 2.1: Autómato para a divisão de orações

Para cada oração, este começa no estado 0. Em seguida, pega em cada palavra da oração e verifica o seu tipo. Se a palavra se tratar de um adjetivo, de um advérbio ou da palavra "tudo", este transita para o estado 1 e guarda a palavra na variável `exp`.

Neste momento, analisa a próxima palavra da oração. Se esta for um adjetivo, um advérbio ou a palavra "tudo", este permanece no estado 1 e adiciona à variável `exp` a palavra que acabou de analisar. Em seguida, analisa a palavra seguinte.

Estando no estado 1, a mudança para o estado 0 só ocorre quando a palavra a ser analisada não é de nenhum dos tipos acima apresentados. Quando a transição do estado 1 para o estado 0 ocorre, significa que a expressão terminou, então o conteúdo da variável `exp` é escrito e reinicializado.

Nos casos em que este se encontra no estado 0 e se depara com uma palavra que não se trata de um adjetivo, de um advérbio ou da palavra "tudo", este despeja o valor da variável `exp` e mantém-se no mesmo estado.

Deste modo, ao fornecermos a frase "Este bolo é muito bom."ao autómato, este vai dividi-la da seguinte forma: ['este', 'bolo', 'ser', 'muito bom'].

É de realçar que as palavras, após serem analisadas, são lematizadas.

2.2.2 2.ª abordagem:

Como foi referido anteriormente, foi atribuído inicialmente um valor de sentimento de ± 0.5 a todas as palavras, mas isto não é uma boa descrição da realidade, já que nem todas as palavras têm o mesmo nível de intensidade. Por exemplo, seria esperado que a frase "Adorei o restaurante."tivesse um valor maior do que "Gostei do restaurante.". Para combater isto, os sentimentos foram todos reavaliados com a ajuda do *dataset* de *tweets* em português trabalhado nas aulas da seguinte forma:

Primeiro, o *dataset* foi truncado, formando um ficheiro com aproximadamente 10000 *tweets* para treino do modelo, e outro ficheiro com 3000 para servir de teste.

Depois foram avaliados todos os *tweets* com o analisador de sentimentos criado e foram reajustados os valores de todas as palavras que aparecem no *tweet* e que estão presentes na forma lematizada no dicionário. Para reajustar os valores, considerou-se que cada *tweet* tinha um valor de sentimento absoluto de 0.7 e somamos a cada palavra 10% do da diferença entre o valor calculado e ± 0.7.

No fim do treino, é feito um teste para determinar se este foi eficaz em que se compara os resultados obtidos com os valores antes e depois do treino. Dos quase 3000 *tweets* analisados:

- 2268 foram avaliados corretamente por ambos os modelos
- 9 foram avaliados corretamente apenas pelo modelo antes de ser treinado
- 675 foram avaliados corretamente apenas pelo modelo treinado
- 34 foram avaliados incorretamente por ambos os modelos

Podemos então verificar que a precisão em termos de detetar se um dado *tweet* é positivo ou negativo aumentou de 75% para 98%. Este valor levanta algumas suspeitas de um possível *overfitting*. Uma razão possível é o facto de todos os *tweets* possuírem um ":)"ou ":(", mas estas não estão a ser incluídos no treino, por isso não justifica o aumento da precisão.

2.3 Recolha da informação do Instagram

Antes de se passar para a implementação da aplicação, construiu-se ainda uma *script* em Python capaz de gerar um ficheiro de texto com todos os comentários de um dado *post* de um determinado utilizador. De modo a enriquecer a aplicação, optou-se por fazer com que esta recolhesse também os conteúdos multimédia do *post*, bem como o número de comentários e de gostos.

Para realizar todas estas operações, esta precisa de receber as credencias de uma conta de Instagram, o nome do utilizador a que pertence o *post* e o seu número. É de realçar que o último *post* publicado é o número 1, o penúltimo é o número 2 e assim sucessivamente. A recolha das informações é efetuada através do módulo Instaloader.

2.4 Aplicação final

Finalizada esta *script*, estavam reunidos todos os componentes necessários para montar a aplicação final.

Tendo conhecimento do nome do utilizador a quem pertence o *post* que queremos analisar e o número do mesmo, esta realiza as seguintes ações:

- Para cada *post* que foi efetuado após aquele que queremos analisar:
  - Chama a *script* que desencadeia o *download* da informação do *post*;
  - Analisa os comentários do *post* utilizando o analisador de sentimentos edificado e calcula o número de comentários positivos, neutros e negativos;
  - Guarda estatísticas do *post* numa lista;
- Calcula o número médio de gostos e do sentimento dos *posts* que foram realizados depois daquele que estamos a analisar. Para além disso, determina a percentagem do número de gostos e do sentimento de cada um destes *posts* comparativamente com os valores médios obtidos;
- Cria um gráfico circular com a percentagem de cada tipo de comentário através do módulo matplotlib;
- Gera uma página HTML com os resultados obtidos.

Abaixo, segue-se um exemplo de uma página HTML gerada pela aplicação:

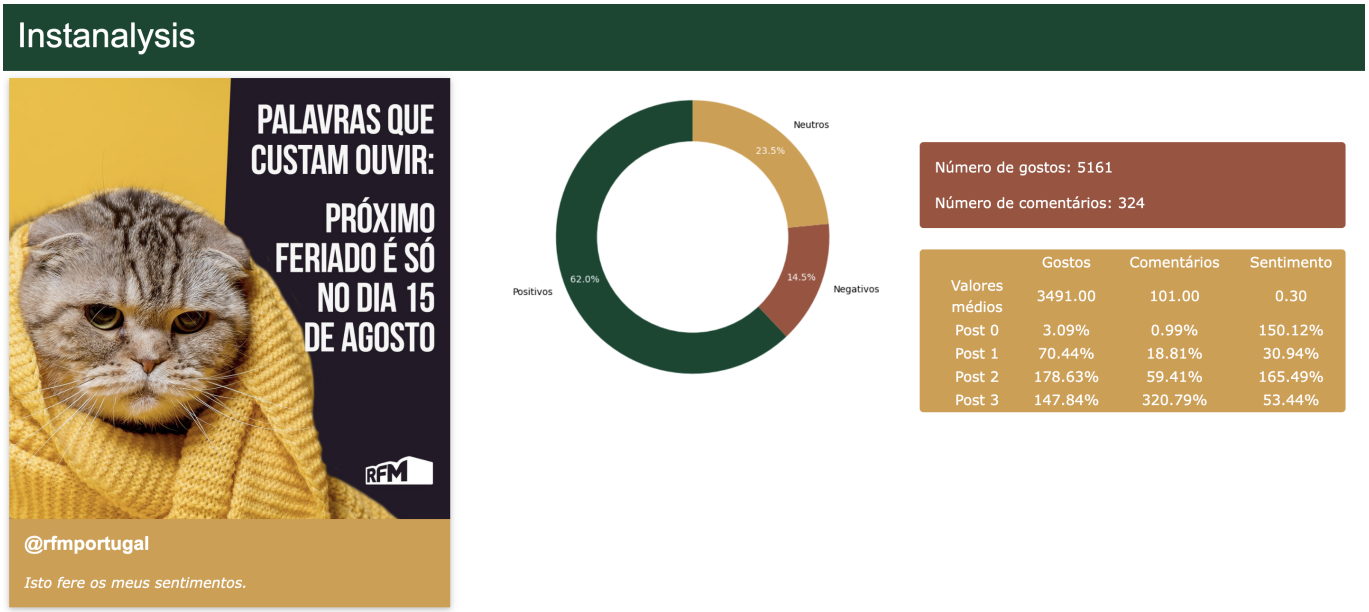

Figura 2.2: Página HTML gerada pela aplicação

Capítulo 3

Trabalho Futuro

Em relação ao trabalho futuro, considera-se que seria interessante aprimorar a aplicação através da criação de uma interface interativa. Deste modo, os utilizadores teriam a oportunidade de selecionarem através da mesma o *post* que gostariam de analisar.

Para além disso, seria interessante expandir o leque de estatísticas disponibilizadas, bem como detetar *text emojis*.

Capítulo 4

Conclusão

No decorrer deste relatório, foi discutido o processo de criação de uma aplicação de analise de sentimentos e a sua implementação no contexto de *posts* de Instagram.

Este projeto permitiu à equipa por em prática muitos dos temas abordados em aula, sendo assim uma excelente forma de consolidar a aprendizagem.

No processo de desenvolvimento, a maior dificuldade encontrada foi definir a forma como seriam avaliadas as orações já que existe um numero infindável de maneiras deferente de o fazer.

Apesar de haver ainda mais informação que podia ser recolhida e apresentada ao utilizador, considera-se que os objetivos iniciais foram cumpridos e a equipa está satisfeita com o resultado final.