Língua Natural – 2º Mini Projeto

Carolina Xavier (81172) & Cláudio Correia (81959)
Grupo 52

Introdução

O objetivo deste projeto é de classificação de texto, com um conjunto de classes definido. Classificação de texto é um problema de processamento de Língua natural (NLP) com o objetivo de atribuir uma (ou mais) classes a um texto.

Neste projeto os textos consistem em perguntas, todas elas centradas no tema de cinema, que devem ser classificadas quanto ao tipo de resposta que lhe deve ser dada. Existem 16 tipos de perguntas (classes): actor_name, budget, character_name, genre, keyword, original_language, original_title, overview, person_name, production_company, production_country, release_date, revenue, runtime, spoken_language, vote_avg.

A classificação é feita com base num ficheiro existente (QuestoesConhecidas.txt), *corpus* de treino que contem várias perguntas já classificadas. A partir do *corpus* ser desenvolvida uma métrica de similaridade que permita classificar uma nova pergunta sobre cinema por comparação com as perguntas classificadas conhecidas.

Proposta de solução

O primeiro passo da solução consiste em fazer um **pré-processamento** do texto de treino e de teste.
O pré-processamento consiste em 4 passos:

1. Remoção de caracteres *newline* '\n' e *tab* '\t';
2. Remoção de pontuação;
3. Substituição de palavras, comuns ao texto e aos ficheiros de recursos, por uma só palavra. Esta substituição é feita na ordem seguinte:
   - list_people.txt - "person"
   - list_genres.txt - "genre"
   - list_jobs.txt – "job"
   - list_companies.txt – "company"
   - list_movies.txt – "movie"
   - list_characters.txt – "character"
4. Passagem de todos os caracteres para letras minúsculas.

Após o pré-processamento das questões efetuamos vectorização, aplicando o algoritmo **TI-IDF**. TF-IDF utiliza *Term Frequency* (TF) e *Inverse Document Frequency* (IDF).
TF consiste em calcular a razão entre o número de vezes que um termo ocorre na questão e o número de termos na questão. IDF permite reduzir as frequências TF de modo a que termos comuns da língua tenham valores mais baixos.

$$TF(termo, questão) = \frac{número \text{ de vezes que um termo ocorre na uestão}}{número\ de\ termos\ na\ questão}$$

$$IDF(termo) = \log\left[\frac{número\ de\ questões}{número\ de\ questões\ em\ que\ o\ termo\ ocorre}\right]$$

$$TF\ IDF(termo, questão) = TF(termo, questão) \times IDF(termo)$$

Para aplicar o algoritmo TF-IDF em *Python* existe o pacote *sklearn.feature_extraction*[1] que contém a função text.TfIdfVectorizer[2], esta função permite vetorizar o texto.
Um termo pode consistir em apenas uma palavra ou em tuplos de palavras consecutivas (N-gramas). Para isso, um dos atributos da função TfIdfVectorizer é *ngram_range* e permite definir os N-gramas a considerar.
Outro atributo da função TfIdfVectorizer é *stop_words* que quando definido como 'english' remove palavras que se presumem pouco informativas num texto em inglês, como "and", "the" ou "him".
Após o cálculo do vetor com as frequências este é normalizado.

```
vectorizer = TfIdfVectorizer(ngram_range=(2,3), stop_words='english')
model = vectorizer.fit_transform(train_questions).toarray()

for i in range(0,len(model)):
      vector[i] = np.true_divide(model[i],np.sum(model[i]))
```

Depois de aplicar o algoritmo TF-IDF calculamos o **cosseno entre vetores** – cosseno entre o vetor da pergunta a classificar e cada um dos vetores das perguntas de treino.
A classe atribuída à pergunta é a classe da pergunta de treino cujo vetor é mais semelhante à pergunta a classificar (maior cosseno).

Resultados experimentais e discussão dos resultados.

Testando a nossa implementação com o ficheiro NovasQuestoes.txt a melhor *accuracy* que obtemos é de 100%, quando efetuamos os 4 passos de pré-processamento, consideramos 2-gramas e 3-gramas e removemos palavras que se presumem pouco informativas num texto em inglês na função `TfIdfVectorizer.`
Sem efetuar o 3º passo o pré-processamento (substituição de palavras comuns ao texto e aos ficheiros de recursos) e remoção de palavras que se pouco informativas num texto em inglês a *accuracy* desce para 66,67%
Quando acrescentamos à fase de pré-processamento de texto a substituição de uma ou mais palavras na pergunta por entidades (disponíveis nos ficheiros de recurso) a *accuracy* aumenta consideravelmente porque a vectorização do texto é feita com base em termos em comum entre as questões, assim quando há por exemplo duas questões formuladas da mesma maneira mas referentes a dois filmes com títulos diferentes estas vão ter vetores mais semelhantes depois de se alterar o titulo dos filmes pela palavra "movie" que passa a ser comum nas duas questões.

Alterando os N-gramas a considerar a *accuracy* desceu para 92,86% com apenas 1-gramas, 95,23% com 1-gramas e 2-gramas. 95,23% com 1-gramas 2-gramas e 3-gramas e 33% com 4-gramas e 5-gramas
O uso de 2-gramas e 3-gramas é favorável à *performance* do algoritmo em comparação com 1-gramas porque fornecem mais contexto ao modelo. No entanto o aumento da ordem dos N-gramas não é benéfico porque passam-se a considerar sequências de palavras demasiado específicas.

Conclusão e trabalho futuro

Depois de explorar os resultados do algoritmo implementado, decidimos manter a solução com a qual conseguimos obter melhores resultados para o ficheiro de teste: efetuar os 4 passos de pré-processamento, considerar 2-gramas e 3-gramas e remover palavras que se presumem pouco informativas num texto em inglês.

A *performance* do algoritmo pode ser melhorada utilizando um maior corpus de treino, que contenha maior diversidade de casos.
No futuro, a implementação do algoritmo de classificação pode ser melhorada aplicando mais alguns mais conceitos de *machine learning*.

Bibliografia

[1] scikit-learn developers (2007-2018). Feature extraction. “http://scikit-learn.org/stable/modules/feature_extraction.html” [Accessed 3 Nov. 2018]

[2] scikit-learn developers (2007-2018). sklearn.feature_extraction.text.TfidfVectorizer “http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html” [Accessed 3 Nov. 2018]