Relatório do Segundo Mini Projeto de Língua Natural 2021/2022

Grupo 40, composto pelas alunas:

- Ielga Oliveira, 92479
- Joana Raposo, 92485

1. Descrição do Modelo

O nosso modelo começa por ler e separar as categorias das questões e respostas relativamente ao conjunto de treino e ao conjunto de desenvolvimento. Nesta separação, no caso do conjunto de treino ficamos com duas listas, em que posições iguais nos dois vetores correspondem ao mesmo par (pergunta-resposta, categoria). No caso do desenvolvimento, o vetor de categorias é inicialmente vazio, e o outro vetor possui o conteúdo das várias perguntas/respostas a serem classificadas. Como pré-processamento, optámos somente por passar o texto a minúsculas. Decidimos assim porque chegámos à conclusão que, por exemplo, remover "stop-words" poderia ser perigoso, já que estamos a classificar frases onde algumas são por si só curtas, o que poderia remover palavras que providenciam imenso significado às frases.

Decidimos também utilizar TF-IDF para transformar o texto em vetores numéricos de modo a sabermos a frequência das palavras presentes no texto, assim fizemos o TF-IDF vectorizer para os dois conjuntos (treino e desenvolvimento). Depois disto, transformamos as listas de perguntas do treino e do desenvolvimento, vetorizados previamente, afim de conterem para cada linha uma lista de números inteiros e a sua importância calculada pelo TF-IDF.

Finalmente, para classificar recorremos ao Support Vector Machine da biblioteca Scikit Learn.

2. Setup Experimental

Para avaliar os vários modelos, guiamo-nos pela taxa de acerto (número de perguntas do ficheiro de desenvolvimento que o sistema classificou corretamente a dividir pelo número total de perguntas e multiplicado por cem).

Como modelo base, fizemos um modelo que como pré-processamento é semelhante ao modelo final: separar as categorias das questões/respostas e passamos todo o texto para minúsculas. Depois, para cada pergunta/resposta do conjunto de desenvolvimento, calculamos a semelhança de Jaccard com todas as frases do conjunto de treino. Selecionamos a frase do conjunto de treino com maior semelhança, e atribuímos à frase que estamos a tentar classificar a mesma categoria que essa frase mais semelhante possui.

3. Resultados

	Resultados (= Precisão resultante de avaliar os modelos com o conjunto de desenvolvimento)
Modelo base	19,6 %
Modelo final	87,0 %

4. Erros na Análise

Verificamos que muitas das respostas erradas dadas pelo nosso sistema correspondiam à categoria de História e Literatura. Este erro é devido à falta de balanceamento dos dados do conjunto de treino. Constatamos que existem mais referências de certas categorias do que outras, por exemplo, existem cento e trinta e oito categorias de História face a apenas quarenta de Geografia. Assim, fica mais provável o nosso modelo classificar as perguntas como certas categorias do que outras, o que introduz um certo enviesamento nos resultados finais.

Adicionalmente, reparamos também que que o sistema algumas vezes devolve História a uma categoria de Geografia e vice-versa, isto devido à semelhança dos dados relativamente a estas categorias como a presença de datas, nomes e localidades.

5. Futuro Trabalho

Se tivéssemos mais tempo para desenvolver o trabalho, gostaríamos de experimentar mais técnicas de pré-processamento ao texto, como lematização, remover a pontuação, etc. Também gostaríamos de, provavelmente, analisar as perguntas separadamente das respostas de forma a explorar se isso melhoraria o sistema. Definitivamente, que também gostaríamos de aplicar classificadores como o BERT, que traria outra dinâmica a este sistema de classificação.

6. Bibliografia

Kanani, B. (2020, April 24). *Jaccard Similarity – Text Similarity Metric in NLP*. Machine Learning Tutorials. Retrieved November 5, 2021, from https://studymachinelearning.com/jaccard-similarity-text-similarity-metric-in-nlp/

Bedi, G. (2018, November 9). *A guide to Text Classification(NLP) using SVM and Naive Bayes with Python*. Medium. Retrieved November 8, 2021, from https://medium.com/@bedigunjit/simple-guide-to-text-classification-nlp-using-svm-and-naive-bayes-with-python-421db3a72d34