Group 21 IPR Project 1 Report

Daniel Correia, Gonçalo Ramos, João Machado

Instituto Superior Técnico

1 Exercise 1

Regarding exercise 1, we used the train subset from the 20newsgroup collection, a part of it, the 100 first documents, to compute it faster. We then created a test input file. In the pre-processing phase, by using the TfidVectorizer we removed English stop-words and also used ngram range between 1 and 3 to get word tokens, bigrams and trigrams. We also used useidf = True to enable IDF re-weighting and thus obtain the TF-IDF scores. To avoid the non-estimation of terms not included in the collection, we decided to estimate TF-IDF scores for all the terms from both the collection and our document. For this, we had to apply the fit function to a joined list of the collection and our document. This way, we learn all the vocabulary, ensuring TF-IDF scores for all candidates. To finish, a sorted dictionary was built, with the reverse variable equal to "True" in order to rank the best 5 keyphrases by their TF-IDF scores.

2 Exercise 2

Regarding exercise 2, we selected the dataset "DUC 2001". We parsed the xml files from that dataset and used a TfidVectorizer, removing stopwords, and with ngram range between 1 and 3 to get word tokens, bigrams and trigrams. We also used useidf = True to enable IDF re-weighting. We performed fit transform of the data parsed to n the vocabulary and IDFs, returning TF-IDF scores. Then we parsed the Json file that contains the most relevant keyphrases estimated by an expert. For each document, we created a dictionary of feature names and TF-IDF scores and selected the top 10 most relevant keyphrases, by sorting the dictionary by descending order of the values (TF-IDF scores). Then we compared, for each document, the keyphrases estimated by the expert with the keyphrases returned by our classifier. We calculated precision, recall and f1 score for each document saving the results in dictionaries. We also calculated the precision@5 metric for each document and followed the same logic to store the values. We calculated the average precision metric for each document and stored the values. After calculating the metrics for each document, we calculated the average values for recall and f1 score. We also calculated the mean value

for Precision@5 metric and the Mean Average Precision. The mean average precision was *0.23* and the mean precision@5 was *0.099*.

3 Exercise 3

Regarding exercise 3, we began by parsing the xml from the previous selected dataset. For each document in the dataset, the candidates were limited either to noun phrases matching a parts-of-speech regular expression or by using the named entity recognition as another test example. After obtaining the candidates with these limitations in each document, for every candidate presented, we calculate the variables to be applied to the both the BM25 and a BM25 combined with the length of the candidate keyphrases to score the candidates. To complete, we compared, for each document, the keyphrases estimated by the expert with the keyphrases returned by our classifier by presenting the mean average precision much like the previous exercise. The mean average precision for noun phrases expression was *0.29* and for named entity recognition was *0.* The value for the named entity recognition was 0 due to none of the keyphrases estimated by the classifier matched the expert keyphrases. Using noun phrases expression we got a higher value of mean average precision than in exercise 2.

4 Exercise 4

Regarding exercise 4, we used the dataset "DUC 2001". First, we parsed the xml files from the dataset and did preprocessing. As an otimization, we performed fit transform of the parsed document list using a TfidVectorizer with ngram range=(1,3) to obtain the feature names. To reduce the number of candidates for a document we only saved as candidates the terms that had a non-zero TF-IDF score. We considered the following features: Position of the candidate in the document, length of the candidate, TF score, IDF score, TF-IDF score, BM25 score for the candidate and its word count. Then for each document, we extracted the features of each candidate, split it into train and test values 50 by 50 and processed them with StandardScaler, to standardize our feature values. We also split our target values into train and test 50 by 50. We trained our classifier Perceptron, with no parameters, with the training values and then tested with our test values and checked our accuracy. Since our accuracy was 0.98 or higher, we decided to train a classifier for each document of the dataset where we didn't split into train and test, but just fit it with our features and target values which corresponds to complete classifier. After that we calculated the confidence scores of all candidates in the document and checked the ones with positive scores (which correspond to the positive classes) and ranked the 5 highest keyphrases. We repeated this process for each document. We then calculated the mean average precision and got *0.* This is because the classifier couldn't estimate the relevant keyphrases for a large part of the documents and the percentage of documents without estimated keyphrases was very high.