Project I Report - Group 1

Francisco Rola 84717 Henrique Almeida 84725 Tomás Oliveira 84773

1 Simple approach based on TF-IDF

As per the project statement, we, in this exercise, implemented an automatic keyphrase extraction method. Firstly, using the *TfidfVectorizer* function, we learned the vocabulary (word tokens, bi-grams, and tri-grams with stop words filtered) and IDF of the recommended corpus, the 20 newsgroups dataset, in conjunction with the document that was selected as the target of the keyphrase extraction method. This document, a BBC news text, was then transformed, using the previously learned model, into a TF-IDF document-term matrix. This matrix, which has one row (BBC document) and numerous columns (terms appearing in the corpus and the document), in order to better rank the extracted terms, was then used to compute the heuristic: TF-IDF $\times$ Term size (number of characters) $\times$ 0.1. After the computation of this heuristic was completed for every term, the 5 terms with the higher score were returned as the document's keyphrases.

It should be noted that the learned vocabulary is comprised of the recommend corpus and the selected document, so that, when the TF-IDF document-term matrix is calculated, if a term that is present on the select document is not on the corpus, a TF-IDF for that term is still computed.

2 Evaluating the simple approach

To evaluate the simple approach, we first retrieved the tokens of the already processed 500N-KPCrowd text dataset. Secondly, as in the previous exercise, we learned the vocabulary of the corpus, in this case, the 500N-KPCrowd dataset, and posteriorly obtained the TF-IDF document-term matrix of the same corpus. Then, the previously used heuristic was applied to the TF-IDF matrix, so that the top-10 ranking terms of each document were retrieved. Moreover, so that the simple approach could be tested, we first extracted the ground truth of the selected dataset and then, using the top-10 candidates per document, and the collection of relevant terms per document computed several metrics with the following results: average precision 23.60%. average recall 5.25%, average F1 8.19%, mean average precision 53.56%, and average precision@5 31.60%. In addition, the precision, recall, and precision@5 for each document were also computed.

3 Improving the simple approach

In this exercise we followed the guidelines provided in the project's statement. We start by parsing through the XML tagged text provided and then apply the chunking pattern provided.

By applying this chunking pattern to the text we obtain a list of candidates to keyphrases which match the said pattern.

Afterwards we create an inverted index using the candidates obtained in the previous step as keys. This inverted structure is the structure we base ourselves on to then calculate the BM25 score of each candidate term (as it provides us with easy access to both document frequency and term frequency). Our final candidate score consists of multiplying the BM25 score by a factor which contains the length of the candidate, by doing so we favour longer candidates (which are less likely to appear elsewhere).

After computing the BM25 score for each candidate we filter the results to only consider the top 10 candidates and that is our final candidate selection. By comparing our results and the ground truth we obtained a mean average precision of 11.50%.

Given the fact that this model should be an improvement on what was done in exercise 1 and 2 we were not satisfied with this result and decided to improve on it. We figured out that the problem with our results was tied to our chunking pattern. The chunking pattern was providing us with candidates whose length often exceeded a tri-gram and therefore would never be present in the ground truth. Additionally, for single words with a single occurrence over all docs the BM25 score would be equal and our sorting by top ten would be too reliant on the random chance they would be placed in the top candidate. We developed two separate and disjoint optimizations which we describe next.

The first optimization was changing the chunking pattern to match the keyphrases present in the ground truth. We developed a simple pattern which considers candidates up to tri-grams of either nouns as well as a single adjective followed by a noun. This optimization led to a mean average precision of 27.89%.

The second optimization involves changing the matching metric (used in the computation of precision), instead of looking for exact matches we check if any term in the ground truth is contained in our candidate and if their Levenshtein distance is less than 10. By doing this we exclude candidates which are very long and potentially contain the whole document and we tackle the issue we observed with the initial pattern which provided us with candidates such as "*big announcement today from phish hq*" while the candidate expected in ground truth was "*big announcement*". Our mean average precision by following this approach was 37.13%.

Combining both optimizations is also possible and provided us with a mean average precision of 69.69%. This value is higher than the 53.56% obtained in exercise 2 which leads us to conclude than an optimal BM25 model can achieve higher results than a simple TF-IDF approach however we rely heavily on the chunking pattern and comparison metric chosen.

4 A supervised approach

In this exercise, we employed a supervised approach to train a perceptron that could differentiate and extract keyphrases from a set of documents. We use the same dataset from previous exercises but split it into two subsets, train and test.

The first step consists in extracting candidate terms from the training data and calculating metrics over them. These can then be used as features to train a perceptron, that assigns a binary decision to each candidate. Before being used, the values are standardized by removing the mean and scaling to unit variance.The main features that have been chosen, due to their contribute for a better classification performance, were the following:

1. TF-IDF: This metric was chosen due to the obtained results in exercise 2.
2. BM25: Identifies the documents that are more significant for a certain term, a relation which can be used to determine the term's importance.
3. Term frequency: By determining the term frequency, we can detect which terms appear more in a certain document, which will likely have a bigger chance of being considered a keyphrase.
4. Document frequency: Terms that appear in more documents will likely have a more significant importance and higher chance of being considered a keyphrase.

Table 1: Mean Average Precision (MAP) for each exercise

	MAP (%)
TF-IDF	53.56
BM-25 (optimizations)	69.69
Perceptron	59.73

5. Document source: This metric maps the theme of a document, available through the file title, to numbers, with the aim of studying its correlation with the type of words in the document, as documents with shared topics might present similar keyphrases.
6. Term length: The presence of terms with a bigger length which also present significant scores for the other metrics are good contenders to be included in the keyphrases set.
7. Position: The position of a term might contribute to uncover its importance as introduction and the first paragraphs will likely contain a larger number of keywords than the rest of the document. One practical example where this happens is in papers or news documents.

After validating that the perceptron was able to provide considerable performance for this classification problem, we then extract the same metrics over the test data, and pass them to the trained perceptron to obtain a prediction of possible keyphrases.The obtained results can then be filtered and reduced to a set of ten keyphrases per document, based on the confidence of the predictions. Finally, we use this set of keyphrases to establish a comparison between this approach and the results from exercise 2, which can be achieved by computing the mean average precision. The best we obtained was a 59.73% Mean Average Precision for the test dataset, which is an improvement over the results from exercise 2, but offering a worst performance than the technique employed in exercise 3 (with optimizations). However, it is important to notice that the selection features was done using only the test data due to the size of the training subset, which leaves rooms for improvements for choosing features that can better transfer knowledge between datasets.

5 Overall Results

To conclude, in table 1 we present the results for all exercises.