Keyword extraction from text
Information Processing and Retrieval course project report

Martin Mirakyan
ist194771
mirakyanmartin@gmail.com

Abstract

This report demonstrates our approach to create a keyword extraction system within the scope of the Information Processing and Retrieval course. We run experiments by testing different approaches of keyword candidate selection and ranking. In the end, we evaluate all the experiments on the test data to show which of those perform the best.

1 Introduction

Keyword extraction systems have a long history of development with different methods and experiments. In the most popular methods the process consists of two steps: candidate selection for the keywords, and candidate ranking after which the top several keywords are selected. This approach has some problems that need to be tackled like how to select the keyword candidates so that we don not miss the important ones and so that we do not have too many irrelevant ones. Or how to rank the candidates after the selection?

We have experimented with different approaches of candidate selection and ranking and demonstrated the empirical evaluation for each at the end.

All the codes developed in the scope of this project are open sourced and available on GitHub [1].

2 Method

In the scope of this project we have experimented with different approaches for keyword extraction. We have used the Inspec [2] dataset for evaluation and training of the system.

[1] https://github.com/MartinXPN/Tecnico/tree/master/InformationProcessing/Project1

[2] https://github.com/boudinfl/ake-datasets/tree/master/datasets/Inspec/

2.1 Exercise 1 - Simple approach

For the first exercise we had to create a simple keyword extraction method that will be trained on a large text corpus and later will extract keywords from a text of our choice. We have trained the extractor on the 20-news grpus dataset [3] and evaluated it on the famous alice.txt [4] text.

The simple approach is based on calculating the TF-IDF scores for each candidate in the collection and then taking the top-5 scored ones, where the candidates are all the unigrams, bigrams, and trigrams of words in the input text.

After running the experiment the extracted keywords for Alice were: alice, said alice, said, hatter, said king.

2.2 Exercise 2 - Evaluation

The next step is to evaluate the approach numerically. The same system with TF-IDF scores and all the unigrams, bigrams, and trigrams from the corpus as the candidates was used as a model. The Inspec dataset was selected for the evaluation. As it has both the target keyphrases for training and testing text collections, we have calculated the Precision, Recall, F1 score, Precision@5, and the average precision for each data sample in the collection. After which the mean values for each were displayed.

	Prec.	Rec.	F1	P@5	MAP
Simple	0.08	0.05	0.06	0.06	0.05

Table 1: Mean values of the metrics for performance evaluation of the simple approach on keyword extraction

[3] http://qwone.com/~jason/20Newsgroups/

[4] https://github.com/mlafeldt/google-python-class/blob/master/basic/alice.txt

2.3 Exercise 3 - Improving the simple approach

To improve the simple approach we have changed the main two parts of the system. Instead of taking all the possible unigrams, bigrams, and trigrams in the document, we have selected only the ones matching a specific regular expression based on part of speech tags. Besides, instead of calculating the TF-IDF score, we have used the BM25 score to measure how good is the candidate.

That change improved the performance a lot. The table below demonstrates the improvements.

	Prec.	Rec.	F1	P@5	MAP
Simple	0.08	0.05	0.06	0.06	0.05
Improve	0.20	0.11	0.13	0.12	0.11

Table 2: Mean values of the metrics for performance evaluation of the simple approach on keyword extraction

2.4 Exercise 4 - Supervised approach

Keyword extraction can also be formulated as a supervised problem. Where we know the target keyphrases and also have some features of the text which can be used to classify whether the candidate is a keyphrase or not. In this approach we have changed only the candidate evaluation step and kept the candidate selection part from the previous exercise. For the candidate evaluation, we have used a well known method called xgboost (Chen and Guestrin, 2016). The input features for each candidates were the TF, IDF, TF-IDF, BM25 scores, and also the number of characters in the candidate. We have treated the problem as a binary classification problem where we classify each candidate as either relevant or not. We have used class weights to balance the classes as there were too many negative classes.

That approach showed comparably similar results to the previous one.

	Prec.	Rec.	F1	P@5	MAP
Simple	0.08	0.05	0.06	0.06	0.05
Improve	0.20	0.11	0.13	0.12	0.11
xgboost	0.21	0.11	0.14	0.10	0.11

Table 3: Mean values of the metrics for performance evaluation of the simple approach on keyword extraction

3 Conclusions and future work

Based on demonstrated results, it is clear that the simple approach is inferior to the one with improved candidate selection and scoring method. Yet, there is a lot of room fore improvements and new experiments.

In the future we could experiment with ensembles of different scores as the classification features, try some more features like the part of speech tag combination of the candidate or its position in the document, etc. The candidate selection step can be improved too by applying a more elaborate method of selection.

References

Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA. ACM.