Keywords Extraction
Machine Learning for Natural Language Processing
2020

Tom Guédon tom.guedon@ensae.fr
Elias Chaumeix elias.chaumeix@ensae.fr

1 Problem Framing

The aim of this project is to extract keywords from a short text. You can find the code on GitHub [1] and Google Colab [2]. For this purpose we used abstracts of research papers and the associated keywords. Our dataset is the Scielo website [3] that we scrapped. It is a Latin American website that publishes articles in English shared by the São Paulo Research Foundation and the Latin American and Caribbean Center on Health Sciences Information. In this project, we chose to combine two different approaches : supervised and unsupervised learning. It seemed interesting to us because it is not usual to have access to the actual keywords of the abstracts, most of the time keywords extraction is dealt with unsupervised learning.

2 Experiments Protocol

In the first part of our project, we run several unsupervised methods such as TF-IDF, Text Rank, Rapid Automatic Keywords Extraction (RAKE) and an adapted Text Rank method using Word2vec, which we will describe later. Each method gives a score to each word, therefore the goal of the second part is to use those weights as input for a supervised algorithm such as a random forest or a logistic regression. The target variable is a binary vector with the size of the number of words of the database and with a 1 if the word is a keyword and 0 otherwise. We choose the F1 score and the precision as metrics in order to compare the different methods. Some methods use extended databases that you can find on our GitHub.

[1] https://github.com/eliaschaumeix45/NLP_Project_3A

[2] https://colab.research.google.com/drive/1rMzOb7v80usva4-_31Io_wzN4-4G9wyX?usp=sharing&fbclid=IwAR05vvI5Lc-YaxSMB6FJXs6J0ZpnzJF6sKemiSwk0dKar3JKkUn4OjkH scrollTo=O1qGpo0yu1Dw

[3] https://scielo.org/en/

3 Results

This is the table of the graph of the reported F1 and precision score :

Method	TF IDF	Text Rank	RAKE	W2v	Random	Rf
F1 score	0.246	0.311	0.138	0.062	0.153	0.297
Precision score	0.293	0.237	0.253	0.245	0.257	0.308

Even though the results do not seem that good, by looking at the confusion matrices, our method is the one that is the best at predicting if a word is a keyword. Our method is also the one that has the lowest false keyword prediction rate which are satisfying and cheering results.
When taking a closer look at the prediction, we realize that when it predicts one word as keyword the word next to it is also given as keyword.
Let's take the pair "time series" as an example. If the word *time* is present more than once it will be likely predicted as a keyword and therefore *series* also. This could be explained by the fact that Text Rank has a high importance in the classification task, and this method is based on measuring the frequences of pairs of words in order to built the weights of each words. Given that our method either predicts both words or none (often), the error can be doubled and that can explain some deceptive results. That is why we could take n-grams into account and assign them a unique score but that would also increase greatly the size of our vocabulary.

4 Discussion/Conclusion

We were glad trying to implement a new method, that also gave us a chance to well understand many unsupervised ones. However, we want to get more deeply into our results and clarify how they could be improved.
First, we faced some lemmatization issues regarding non English words, names and surnames that were pretty frequent. Also some keywords are not in the abstracts which make it impossible for the algorithm to predict them regarding our methods. As said before, we have had issues concerning bi-gram and tri-gram keywords. Then, as we aggregate several methods, it is difficult to work on a universal vocabulary. Therefore the most obvious solution was to only consider one-gram item. Therefore, the bi-gram and tri-gram keywords were split in one-gram items.

Now let's introduce some extensions that could be done to improve the results. The first thing would be to take into account the n-grams. Another point is that we use machine learning algorithm as if our data were independent. This assumption does not seem to be verified as the words are part of abstract. Therefore using mixed effects model to take into account the covariance structure of our data could be interesting. (This can be difficult with python because it does not provide a good library to do that compared to R).
Finally adding some unsupervised methods in order to have more features could improve the results, we thought about YAKE! or BERT.