Clustering of scientific papers using Natural language processing

Léo Kazmierczak, Clément Malindi

Abstract

Document classification is one of the fields that has been completely renewed lately, combining natural language processing and machine learning techniques. In this work, in order to classify scientific documents we focus on two particular clustering methods, the K-means algorithm and the Ward method using as a single feature : the TF-IDF, feature used in natural language processing. Several measures have been used to characterise the performance of these algorithms for the given problem.

Contents

1. INTRODUCTION

Clustering describe the process of organizing objects into groups whose members are similar with respect to a similarity or distance criterion.

The idea of this work is to extract the main subject of a scientific paper from the metadata,and then use a clustering process, in order to classify these papers within clusters.

1.1 Related works

[2] The original inspiration of this work is the paper of Ed Collins, Isabelle Augenstein and Sebastian Riedel. In their work they tried to summarize scientific papers to restraint them to his main arguments. A a data set they used a large resource of author provided summaries which will be the same data set that we will use. [5] Beumer investigated text document clustering applying the k-Means algorithm to three different data sets using different distance measures and data characteristics such as using the BoW representation approach, TF-IDF in combination with the cosine or PPMCC metric. [6] Sumayia Al-Anazi et al used several clustering models for graduation project documents at King Saud University. Three cluster similarity measures were tested and the quality of the resulting clusters was evaluated and compared.

1.2 Problem description

In this paper, we want to explore clustering algorithms in order to classify scientific papers, just using their description. As it is unsupervised data, we don't have a prior idea on the topics covered in those articles. So the challenge is not just to classify these articles in clusters, but also to find relevant measures and information to describe the performance of the clustering process.

2. Tools and techniques

2.1 Data mining

For the data set, at first we wanted to use the same one as [3]. But it seems that the code of the Data Downloader tool on their github is out of date. So we had to build ourselves the Data Downloader. To do so, we made a parser that is using the http://www.sciencedirect.com/ API to download all the papers referenced in the [2] paper. This downloader is downloading XML files with all the metadata of the papers. To accelerate the process we used threads because downloading 10000+ paper one by one is a very long task. Downloading one paper is a task of about 2 secs, downloading all the papers without threading our program would take approximately a 6 hours long task. The code of the up to date downloader is available on our github. Then when we had downloaded all the papers we had to build a parser that would extract the fields that interest us to classify the papers. To do so we used the python Library called xml, we extracted the following fields : the title, the abstracts and the given subjects in the metadata.

2.2 Methods used

2.2.1 Text Preprocessing

The data that we get in the Data mining section has a lot of irrelevant information and is not exploitable as is. In order to apply clustering process, we first have to clean the data, in Natural Language Processing, these steps are generally :

- ***Word Tokenization:*** This step aims at dividing the text into smaller units and by the way remove punctuation. In the case of this study, we used word tokenisation, each document was split into a sequence of words, along with its title.
- ***Stemming:*** This process aims at reducing inflected or derived words to their word stem, base or root form—generally a written word form. The stem need not be identical to the morphological root of the word, the idea here is to reduce the number of words in the corpus and simplify the process.
- ***Stop words removal:*** Stopwords are words that are commonly used in different types of text, these are removed from the documents as they do not add relevant information for the process. These words have to be carefully chosen regarding the project.

2.2.2 Features extraction

In order to extract relevant information from the corpus, we used a well known feature in NLP, which is the term frequency-inverse document frequency (**TF-IDF**), this feature reflects how important a word is important in a corpus of documents. Words that are frequent in a document but not across documents tend to have high TF-IDF score (cf Figure1). As we compute TF-IDF for every word in each document in the corpus, a matrix of shape Number of Documents*Number of words can be create.

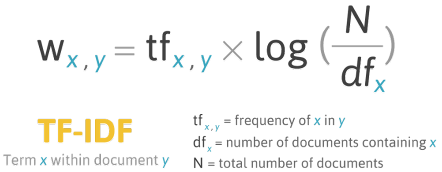

$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$
TF-IDF
Term x within document y
$tf_{x,y}$ = frequency of x in y
df_x = number of documents containing x
N = total number of documents

Figure 1. Mathematical Definition of TF-IDF (from [4])

2.2.3 Clustering Methods

We used three different clustering methods:

- ***K-means clustering:*** K-means clustering is an unsupervised machine learning algorithm. The goal of this algorithm is to find K groups in the data. The algorithm works iteratively to assign each data point to one of K groups based on the features that are provided. Data points are clustered based on feature similarity.
- ***Ward method:*** Ward's method starts with n clusters, each containing a single object. These n clusters are combined to form a cluster containing all objects.At each step, the process creates a new cluster that minimises the variance, measured by an index called E function of the sum of squares.

2.3 Metrics used to validated results

In order to evaluate the relevancy of the clustering process, we have to use some appropriate measure.

- ***Silhouette score:*** The silhouette score is a measure for evaluating a clustering process. It indicates how similar an object is to its own cluster compared to other clusters. Its value varies from -1 to 1. A value close to 1 means that the clustering pattern is relevant and a value close to -1 means that it is not.
- ***Publication histogram*** As the dataset is unsupervised, the data is not labelled, but we had access to some metadata that can help characterise the articles. Thus, in this work, we used the publication of each article in a given cluster to see if a set of publications can be characteristic a cluster.

- ***Neighbouring score*** This measure indicates, for a given document, the number of its p nearest neighbours that are in the same cluster. To consider the neighborhood of a given document, we used the cosine similarity on the TF-IDF matrix (cf figure 2).

$$similarity(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Figure 2. Mathematical Definition of Cosine Similarity

3. Experiments and results

3.1 Dataset

As explained above, the dataset contains the description of 10,000 articles with their title, publication and keywords. In a first step, we used 1000 articles to test the clustering algorithm and in a second step, we used 6000 articles out of the 10000. The article are from 100 different publications of various scientific fields.

3.2 Methodology

- ***Preprocessing:*** Once the database (1000, or 6000) articles was extracted, all the preprocessing steps explained above were applied. We also had to choose a relevant set of empty words, as we only deal with scientific articles, so in addition to the classical set of stop words provided by nltk (be, have, for, and, ...), we added words that often appear in scientific articles (Data, Result, Problem, Model, ...). ...).

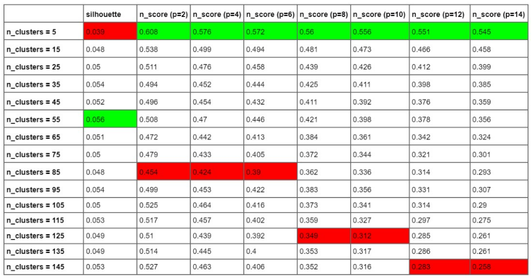
- ***Feature extraction:*** The only feature used in this work is the inverse document frequency (Term-frequency). The output is a matrix of the form Number of documents*Number of words, where each coefficient corresponds to the tf-idf coefficient for a given document and a given word.

- ***Clustering process:*** The clustering process have been applied independently using the scikit learn library. Their performances have been evaluated for different parameters. As an input of these algorithm, we do not use directly, the TF-IDF matrix, but a distance matrix computed using cosine similarity.

3.3 Results

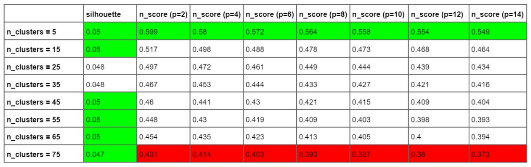
3.3.1 K-means

The only parameter here is the number of clusters, so the idea is to analyse the performance of the clustering process when the number of clusters varies.



	silhouette	n_score (p=2)	n_score (p=4)	n_score (p=6)	n_score (p=8)	n_score (p=10)	n_score (p=12)	n_score (p=14)
n_clusters = 5	0.039	0.608	0.576	0.572	0.56	0.556	0.551	0.545
n_clusters = 15	0.048	0.538	0.499	0.494	0.481	0.473	0.466	0.458
n_clusters = 25	0.05	0.511	0.476	0.458	0.439	0.426	0.412	0.399
n_clusters = 35	0.054	0.494	0.452	0.444	0.425	0.411	0.398	0.385
n_clusters = 45	0.052	0.496	0.454	0.432	0.411	0.392	0.376	0.359
n_clusters = 55	0.056	0.508	0.47	0.446	0.421	0.398	0.378	0.356
n_clusters = 65	0.051	0.472	0.442	0.413	0.384	0.361	0.342	0.324
n_clusters = 75	0.05	0.479	0.433	0.405	0.372	0.344	0.321	0.301
n_clusters = 85	0.048	0.454	0.424	0.39	0.362	0.336	0.314	0.293
n_clusters = 95	0.054	0.499	0.453	0.422	0.383	0.356	0.331	0.307
n_clusters = 105	0.05	0.525	0.464	0.416	0.373	0.341	0.314	0.29
n_clusters = 115	0.053	0.517	0.457	0.402	0.359	0.327	0.297	0.275
n_clusters = 125	0.049	0.51	0.439	0.392	0.349	0.312	0.285	0.261
n_clusters = 135	0.049	0.514	0.445	0.4	0.353	0.317	0.286	0.261
n_clusters = 145	0.053	0.527	0.463	0.406	0.352	0.316	0.283	0.258

Figure 3. Silhouette score and Neighbouring Score (n_score) for the K mean method on about 1000 documents, for different number of clusters (n_clusters)



	silhouette	n_score (p=2)	n_score (p=4)	n_score (p=6)	n_score (p=8)	n_score (p=10)	n_score (p=12)	n_score (p=14)
n_clusters = 5	0.05	0.599	0.58	0.572	0.564	0.558	0.554	0.549
n_clusters = 15	0.05	0.517	0.498	0.488	0.478	0.473	0.468	0.464
n_clusters = 25	0.048	0.497	0.472	0.461	0.449	0.444	0.439	0.434
n_clusters = 35	0.048	0.467	0.453	0.444	0.433	0.427	0.421	0.416
n_clusters = 45	0.05	0.46	0.441	0.43	0.421	0.415	0.409	0.404
n_clusters = 55	0.05	0.448	0.43	0.419	0.409	0.403	0.398	0.393
n_clusters = 65	0.05	0.454	0.435	0.423	0.413	0.405	0.4	0.394
n_clusters = 75	0.047	0.431	0.414	0.403	0.393	0.387	0.38	0.373

Figure 4. Silhouette score and Neighbouring Score (n_score) for the K mean method on about 6000 documents, for different number of clusters (n_clusters)

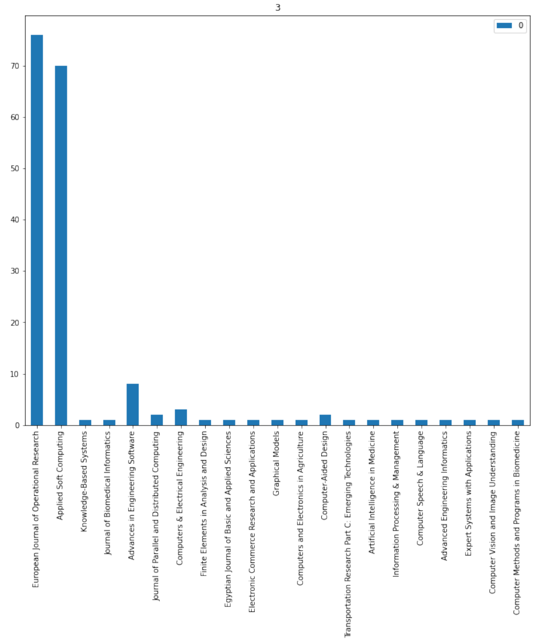


Figure 5. Some Publication Histogram obtains with the K means (n_clusters = 5) and for 1000 articles

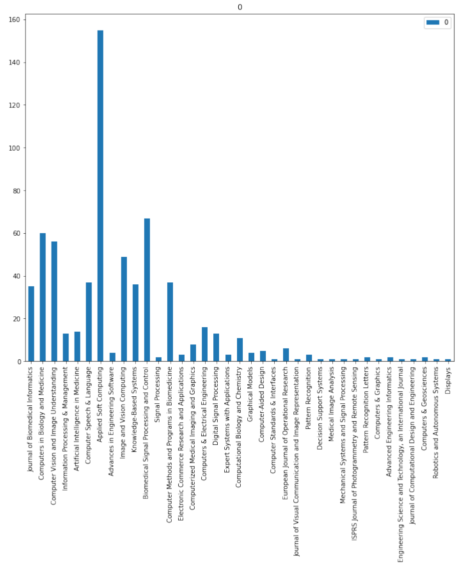
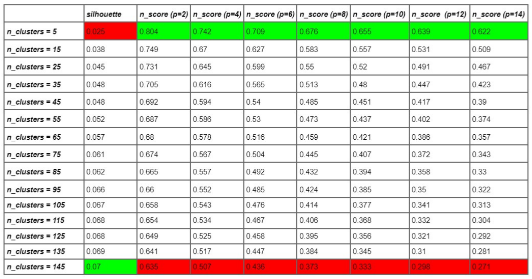


Figure 6. Some Publication Histogram obtains with the K means (n_clusters = 5) and for 6000 articles

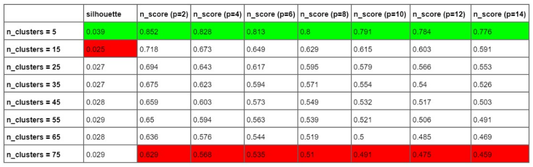
3.3.2 Ward method:

As well as the k means method, the only parameter to change, is the number of clusters.



	silhouette	n_score (p=2)	n_score (p=4)	n_score (p=6)	n_score (p=8)	n_score (p=10)	n_score (p=12)	n_score (p=14)
n_clusters = 5	0.025	0.804	0.742	0.709	0.676	0.655	0.639	0.622
n_clusters = 15	0.038	0.749	0.67	0.627	0.583	0.557	0.531	0.509
n_clusters = 25	0.045	0.731	0.645	0.599	0.55	0.52	0.491	0.467
n_clusters = 35	0.048	0.705	0.616	0.565	0.513	0.48	0.447	0.423
n_clusters = 45	0.048	0.692	0.594	0.54	0.485	0.451	0.417	0.39
n_clusters = 55	0.052	0.687	0.586	0.53	0.473	0.437	0.402	0.374
n_clusters = 65	0.057	0.68	0.578	0.516	0.459	0.421	0.386	0.357
n_clusters = 75	0.061	0.674	0.567	0.504	0.445	0.407	0.372	0.343
n_clusters = 85	0.062	0.665	0.557	0.492	0.432	0.394	0.358	0.33
n_clusters = 95	0.066	0.66	0.552	0.485	0.424	0.385	0.35	0.322
n_clusters = 105	0.067	0.658	0.543	0.476	0.414	0.377	0.341	0.313
n_clusters = 115	0.068	0.654	0.534	0.467	0.406	0.368	0.332	0.304
n_clusters = 125	0.068	0.649	0.525	0.458	0.395	0.356	0.321	0.292
n_clusters = 135	0.069	0.641	0.517	0.447	0.384	0.345	0.31	0.281
n_clusters = 145	0.07	0.635	0.507	0.436	0.373	0.333	0.298	0.271

Figure 7. Silhouette score and Neighbouring Score (n_score) for the Ward method on about 1000 documents, for different number of clusters (n_clusters)



	silhouette	n_score (p=2)	n_score (p=4)	n_score (p=6)	n_score (p=8)	n_score (p=10)	n_score (p=12)	n_score (p=14)
n_clusters = 5	0.039	0.852	0.828	0.813	0.8	0.791	0.784	0.776
n_clusters = 15	0.025	0.718	0.673	0.649	0.629	0.615	0.603	0.591
n_clusters = 25	0.027	0.694	0.643	0.617	0.595	0.579	0.566	0.553
n_clusters = 35	0.027	0.675	0.623	0.594	0.571	0.554	0.54	0.526
n_clusters = 45	0.028	0.659	0.603	0.573	0.549	0.532	0.517	0.503
n_clusters = 55	0.029	0.65	0.594	0.563	0.539	0.521	0.506	0.491
n_clusters = 65	0.028	0.636	0.576	0.544	0.519	0.5	0.485	0.469
n_clusters = 75	0.029	0.629	0.568	0.535	0.51	0.491	0.475	0.459

Figure 8. Silhouette score and Neighbouring Score (n_score) for the Ward method on about 6000 documents, for different number of clusters (n_clusters)

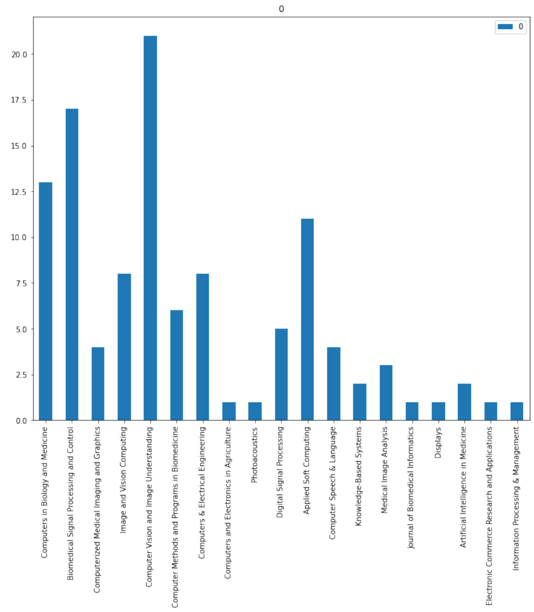


Figure 9. Some Publication Histogram obtains with the Ward Method (n_clusters = 5) and for 1000 articles

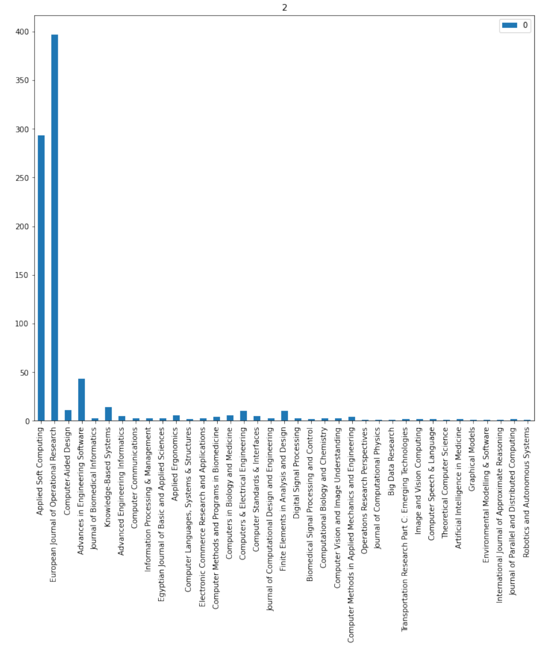


Figure 10. Some Publication Histogram obtains with the Ward Method (n_clusters = 5) and for 6000 articles

3.4 Discussion

The two methods used in this work are quite similar as they both require a prior estimation of the number of clusters, and in a way, they also have similar characteristics regarding their performance. For example, when the number p of neighbours to be considered in the n_ score increases, the n_ score globally decreases (cf Figure 3, Figure 4, Figure 7 Figure 8), which seems logical because if there are more documents to be considered, then they are more likely to be in a different cluster because they are more distant from each other.

And regarding the histogram of publications, we can notice that both clustering methods create clusters with the same type of profile, either some publications are very present (cf. Figure 5 and 10), which is a proof of a good process because it means that the documents in the same cluster tend to be from the same publication, or the distribution of publications is rather flat (Figure 6 and 9), but the publications in question deal with the field, Biomedicine, Biology and Computer Vision for example, which also reflects a good clustering.

These methods also show some particularities in their performance. Regarding the silhouette score, even if for both techniques, their score is very close to 0 which means that the distance between clusters is not significant, we can still notice some differences. First of all, contrary to the Ward method, when we increase the number of clusters, the silhouette score does not vary so much for the k-means, either for 1000 or 6000 items. We can also see that the silhouette score increases with the number of clusters, for the ward method, this is due to the fact that if there are not enough clusters, the intra-class inertia cannot be minimized which is the goal of the ward method, indeed by increasing the number of clusters, it is easier to minimize this inertia.

We can also see that Ward's method has better results for the n-score than the K-means algorithm.This observation seems logical because at each step of the Ward process, the two closest documents or groups of documents are linked together to form a new cluster. Using this process, it is much more likely that two close documents are in the same cluster, than using the K-means method, especially when there is a large number of clusters. Indeed, if two centroids are close, which can be the case if there is a large number of clusters, then two documents even if they are close can be in two different clusters.

4. Conclusion

To conclude, in this paper we used two different hierarchical clustering methods: The **K-means clustering** and the **Ward Method**, in order to cluster scientific articles. To do this, we used features well known in natural language processing, namely the **TF-IDF** which reflects the importance of a word in a corpus of documents. The two models have quite different performances, indeed, K-means performs better for the silhouette score (clusters are better separated), and Ward's method outperforms K-means for the n-score (documents in the same cluster are closer). Thus, depending on the application, either method can be used, if the application needs well-separated clusters, then the K-means algorithm is more suitable, but if it needs more precise clusters, then the Ward method should be used. One of the disadvantages of this type of algorithm is that it requires a lot of resources and can be very slow to compute, which is why it is sometimes necessary to stop the algorithm even if there has been no convergence, especially if the number of data is large, which is why the study with 6,000 documents admits more uncertainty than the other.

References

**[1] SciPDF Parser :
https://github.com/titipata/scipdf_parser**

**[2] A Supervised Approach to Extractive Summarisation of Scientific Papers
https://arxiv.org/abs/1706.03946**

[3] Dataset builder:
https://github.com/EdCo95/scientific-paper-summarisation

[4] Ashutosh Bhardwaj, ”Silhouette Coefficient” url : https://ted-mei.medium.com/demystify-tf-idf-in-indexing-and-ranking-5c3ae66c3fa0

[5] Lisa Beumer, ”Evaluation of Text Document Clustering Using K-Means” url : https://dc.uwm.edu/cgi/viewcontent.cgi?article=3354context=etd

[6] Sumayia Al-Anazi, Hind AlMahmoud, and Isra Al-Turaiki. “Finding Similar Documents Using Different Clustering Techniques”. In: Procedia Computer Science 82 (2016). issn: 1877-0509. doi: https://doi.org/10.1016/j.procs.2016.04.005. url: http://www.sciencedirect.com/science/article/pii/S1877050916300199.