Information Processing and Retrieval

Part 1 Project Report

Group 08:
Daniele Avolio, ist1111559
Michele Vitale, ist1111558
Luís Dias, ist198557

1 Problem statement

In the same dataset context as Part I of this project, we are now addressing the tasks of clustering and classification with both an unsupervised approach and a supervised one, using summaries that are being provided for each document as reference.
Tasks conducted in this second part are:

- **Part A: Clustering;** for each document, the goal is grouping sentences based on their features and similarities. With this done, it is easy to select the most relevant sentences based on some criteria and algorithm that we defined.
- **Part B: Classification;** given a document, the goal is to split it into sentences and, using a binary classifier, define wether each sentence belongs to a summary or not.

This report can not contain all the data and graphs that we produced, so for more complete informations it is strongly suggested to check the comments on the provided notebook.
Some tasks were again very intensive in term of computation, so we decided to not use BERT embedding representations, since it would increase by a lot the time needed to run the code. Thus, our attention was on space representation using TF-IDF.

2 Adopted solutions

Part A: Clustering

This part is conducted in an unsupervised approach, with the idea of grouping sentences with clustering algorithms. In particular, we used the **sklearn** library, with the AgglomerativeClustering class as suggested.
The main paths to explore in this part are:

- **Number of clusters and used metrics:** the main challenge here was the correct choice of the **number of clusters**, because it's a parameter than could have a big impact on the results. Using **silhouette score** we have solved this problem in a iterative way.
- **Sentences selection:** the second challenge was to select the sentences that would be used to build the summary. Using **centroids** of each clusters, we were able to build summaries that had more topics and were more representative of the original text.

Number of clusters and used metrics

A problem related to part A is to define a good number of clusters to represent the feature space of the document.
In this part, we tried to construct a custom metric taking into account Silhouette score, Calinski-Harabasz score and a function of the number of clusters. The idea was to consider the number of clusters, that can vary in the range $[2, numberOfSentences]$, to avoid high sparse clusters representation with single-sentence clusters, but in the end we noticed that the silhouette score by itself was performing overall better.
We are leaving the code for the custom metric in the notebook, but it is commented since it was not used in the final version of the code.

Sentences selection

Another relevant problem, once completed the clustering part of the project, was to decide the criteria to pick the sentences to use in the summarization. We decided to use the **centroid** of each cluster and, based on the distance from it, we pick the required number of sentences. We have done some tests on different criteria, but others ideas that we had were not really convincing on summaries, so we kept the centroid distance as metric.
It is important to note that this strategy has a limitation, since with high numbers of sentences picked from each cluster there might be some redundancy in summaries. Nevertheless, with a low number of picked sentences it is leading to convincing results, taking into the summary relevant sentences.

Part B: Classification

In this section the task is to create a binary classifier that has as goal to discern if a sentence should belong to the summary or not. We used models from the **scikit-learn** library, and in particular the machine learning techniques that we exploded are Random Forest, Gradient Boosting, Gaussian Naive Bayes, K-Neighbors and Multi-layer Perceptron.
Before the training phase, the challenge was to find the best features and the correct shape to represent data and build the models. We ended up with the following structure.

similarity	n_sentence	n_words	n_stopwords	n_keywords	length_of_sentence	tfidf_score
position_in_doc	category	n_nouns	n_verbs	n_adjectives	n_adverbs	id
summary(target)						

We then analyzed the features with a correlation matrix, the Shapley value of each features via the **SHAP** library and the Scikit-learn built-in feature importance. This process lead us to drop some features, with the following schema being the one used to train our models.

similarity	n_sentence	n_words	n_stopwords	n_keywords	length_of_sentence
tfidf_score	position_in_doc	n_nouns	n_verbs	n_adjectives	n_adverbs
category_business	category_entertainment	category_politics	category_sport	category_tech	Summary(target)

At this point, we could train our models and evaluate them with the common machine learning evaluation metrics.

3 Proposed questions