Classifying newspaper articles using reader's comments Machine Learning for Natural Language Processing 2021

Fauchon Alexandre
ENSAE
alexandre.fauchon@ensae.fr

Moutachaker Inès
ENSAE
ines.moutachaker@ensae.fr

1 Problem Framing

We work on sequence classification and more precisely on text categorization. We aim at predicting the category / class of newspaper articles (National, Foreign, Culture, Sport...) based on the compilations of their respective comments.

This analysis is based on the 'New York Times Comments' dataset[1], which gathers online comments to New York Times articles. The category of each commented article is provided. As the dataset is labelled, we can resort to supervised learning tools.

We explore different approaches so as to assess and compare their performance.[2]

2 Experiments Protocol

Our analysis followed the steps below:

1. Data preparation: we compiled comments made for the same articles, and we checked that categories of commented articles were balanced. We decided to drop under-represented categories and to implement data reduction of an over-represented category.

2. Exploratory analysis: we explored the data so as to grasp some patterns and to see whether some words are more frequent in some fields/subjects. We used words clouds.

3. TF-IDF (Term Frequency / Inverse Document Frequency) approach: the idea of this approach is that rare words are the ones encompassing the information. We converted texts into DTM (document term matrix) and used the TF-IDF metric:

$$w_{x,y} = tf_{x,y} * \ln\left(\frac{N}{df_x}\right) \tag{1}$$

This metric associates a score to each word or bigram (combinaison of two words) in a given pool of documents: $w_{x,y}$ is the TF-IDF metric associated to a given word x in document y, $tf_{x,y}$ is the frequency of the term x in the document considered y and df_x is the number of documents that contain x in the whole dataset. $tf_{x,y}$ measures how common is a word in a document and $\ln\left(\frac{N}{df_x}\right)$ how rare - then informative - it is.

We trained different algorithms and found that the logistic regression had the best performance. Classes are overall not mistaken for other, at least not systematically for close topics.

4. We trained a recurrent neural network. The architecture is heavily based on Aakanksha NS[3]'work :

· Each compilation of comments is tokenized. Then, each token is transformed into its index-based representation. We restricted the size of sequences to 1000 tokens (actually, only 25% of compilations of comments have less than 1000 tokens, but going beyond this threshold would have slowed down the execution).
· Embedding layer : each index-based representation is passed sequentially through an embedding layer based on glove vectors

[1]https://www.kaggle.com/aashita/nyt-comments

[2]Google colab : https://colab.research.google.com/drive/1RM6GupxYvugHj3J9oAqLU8L_mwoSZxB_#scrollTo=eKvNECngyBiY

[3]Aakanksha NS "Multiclass Text Classification using LSTM in Pytorch : predicting item ratings based on customer reviews" (https://towardsdatascience.com/multiclass-text-classification-using-lstm-in-pytorch-eac56baed8df)

(50d).
· LSTM layer : as LSTM models have usually good performance working with long texts, we chose a LSTM layer.
· Linear layer : it produces an output whose size is equal to the number of categories / classes.

In order to train the neural network, we used a cross-entropy loss, which is currently used when dealing with multi-class classification tasks. Cross-entropy loss increases as the predicted probability diverges from the actual label. The training was made on mini-batches of size 25.

Note that we trained this neural network on a bigger dataset than the other models. Indeed, after first very disappointing results, we guessed that the training set might me too small, and enlarging the dataset did improve the results by a lot.

5. Eventually, we used another embedding: word2vec and LDA. The idea was to first assess the performance of each of these methods and then to explore the use of a mixed approach combining the two. We used also a logistic regression and compare the results with the TF-IDF approach.

   LDA analysis consists in grouping documents in a predefined number of classes. Each class is modelled by a mix of theme linked to the words used in the document. This method relies on the two following assumptions: documents consists in a mixture of topics. Each of them belongs to each of the topics to a certain degree. Each topic is a generative model which generates words of the vocabulary with certain probabilities. Words frequently occurring together will have more probability

6. Evaluation: for each model, we assessed the performances of our models through one or several of the following methods : computation of accuracy, use of confusion matrices, plot of the ROC curve (quantitative evaluation), reading of articles that were misclassified (qualitative evaluation).

3 Results

The performances depends a lot on the method.

The LSTM approach offers good (about 40% of accuracy) but rough classification: the confusion matrix reveals that close categories are misclassified and mistook for one another. For example, ”Editorial”, ”OpEd”, ”National”, ”Foreign”, which deal a lot with politics, are mixed up a lot. The performance is better for leisure-related categories (”Culture”, ”Dining”...): they are identified from each other and from politics-related categories. Regarding qualitative evaluation, we had a look at articles that were misclassified. Most of the time, it makes a lot of sense : an ”OpEd” article is classified as ”Sports” because it deals with baseball players, a ”Weekend” article is classified as ”Culture” because it deals with movies... A lot of those mistakes could have been made by humans.

The TF/IDF is performing quite well combined with a logistic regression. This can be explained by the fact that rare words are given more importance for the prediction. As they might be the one that enable to disentangle closely related topics, this explains why we do not observe “clusters” of closely related topics.

The last approach using both word2vec and LDA shows that combining the two models gives better performances. We still in this case experience a “clustering” of closely related topics. This can be explained by the fact that some words, when isolated are not relevant for a theme but make sens when they are associated with other words: a “word-sense disambiguation“ issue.

4 Discussion/Conclusion

This short study shows that topic classification using online comments is better using TD/IDF approach. However, if the available dataset are large, neural networks (LSTM) can provide interesting performance.

To expand this study, we provide mainly two ideas. One is to implement a lemmatizer before the TF-IDF so as to see whether it can improve the performances. In some way, it should as

Another option could be to fine tune the model, we can think of weighting the penalisation, giving more importance to errors between close topics in order to avoid the clustering effect we observed. This may be done using a F_β approach, or by identifying clusters of closely related topics.