# *Political Programs Classification*
# Machine Learning for Natural Language Processing 2021

**Paul Couairon**
ENSAE
paul.couairon@ensae.fr

**Loïc Jourdan**
ENSAE
loic.jourdan@ensae.fr

## Abstract

The permanent debate of ideas that confront the different components of the political life, enlightens their disagreements about numerous current stakes and their priorities. For some writers (Piet et al., 2015), one can define the political priority of a political party as the relative weight of a theme in its political program. For example, it is reasonable to expect the green candidates to focus on environmental issues whereas a socialist candidate would be more inclined to deal with social questions. A thorough analysis of such programs bringing together the parties' different engagements and themes must then measure their ideological divergences as well as their evolutions. This is precisely the goal of the *Comparative Manifesto Project* which gathers the elements suggested by most of the parties that run for election at legislative elections. A program is cut into sense units and each of them is manually annotated. Through this project, we aim to automate this time-consuming annotation process.

## 1 Problem Framing

The data collected by the *Comparative Manifesto Project* form the basis of this political programs classification project. Each sense unit is attributed to a code which reads as a three digits number. The code's first digit refers to the sentence's main theme. For example, a sentence whose code's first digit is 1 deals with "External Relations", the digit 2 deals with the theme "Freedom and Democracy" and so on.We have then seven different main categories which are themselves subdivided into smaller categories represented with the second and third digits of the code. Thereby, the goal we set is to automatically infer the first digit of each sentence with the annotated data the *Comparative Manifesto Project* provides, via Natural Language Processing methods. With regards to the abundance of data available, we chose to restrain our analysis framework on the UK's elections between 2015 and 2019. On the one hand, the data are in english which is practical to check some results. On the other hand, although the considered time period is short, it offers enough data in order to lead a consistent analysis. However, we note that the UK's agitated political climate at this period (Brexit) may have influenced the analysis, the speeches varying along with the stakes the political leaders had to face.

## 2 Experiments Protocol

The work we led[1][2] was divided into three key parts. The first one, which is also the most important in order to ensure the good behavior of future algorithms, is the preprocessing of data and some descriptive statistics. For each sentence in the dataset, we dropped the stop-words and we lemmatized the vocabulary to work with the canonical form of words. Posterior to this crucial step, we took care to perform a thorough descriptive analysis of the dataset so as to get a precise representation of the data we manipulate and to anticipate the prospective behavior of the models. We have then observed the vocabulary's distribution: What fraction of the vocabulary represents what percentage of the words? What are the 10 most frequent words in each category? What is the distance between categories' dictionary? Plus we paid extra attention at the classes repartition which might be a dangerous pitfall for our models.

Eventually we tried to manually classify a sample of sentences in the database in order to evaluate the difficulty of the task we tried to automate. It turned out this task is a lot more complex than it looks due the proximity between some categories.

[1] Google Colab Notebook: shorturl.at/bqBGU
[2] GitHub: shorturl.at/opNTX

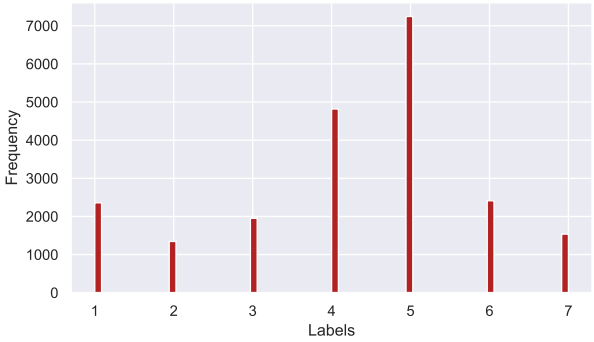


Figure 1: Labels repartition

The remaining work concerns the implementation of NLP algorithms. We decided to divide this bit into two parts. First we wanted to use different machine learning algorithms trained on different embedding methods to have a first glance at the interesting metrics we have to consider in order to evaluate the quality of the classification. Finally we used deep neural networks with an LSTM architecture which capture long-term dependencies within sentences and at last Universal Sentence Encoder (USE).

## 3 Results

The automatic labeling task seems difficult according to descriptive statistics and previous manual classification. We present below a table summarizing the results we obtained on a *3-fold cross-validation* running different machine learning algorithms (SVC, Multinomial Naives Bayes, Bernoulli Bayes, Extra Trees Classifier) on different types of word representations (Bag-of-Words, TF-IDF, Word2Vec).

| | Accuracy | Recall | Precision | F1-Score |
|---|---|---|---|---|
| **SVC TF-IDF** | 0.628 | 0.630 | 0.627 | 0.624 |
| **Mult NB** | 0.619 | 0.617 | 0.612 | 0.605 |
| **Bern NB** | 0.591 | 0.594 | 0.592 | 0.586 |
| **SVC** | 0.586 | 0.583 | 0.596 | 0.559 |
| **XTrees Glove** | 0.536 | 0.534 | 0.564 | 0.494 |
| **XTrees Glove TF-IDF** | 0.531 | 0.530 | 0.560 | 0.486 |
| **XTrees W2V TF-IDF** | 0.475 | 0.478 | 0.462 | 0.447 |
| **XTrees W2V** | 0.468 | 0.469 | 0.460 | 0.441 |

Figure 2: ML models performances

The results displayed here are quite heterogeneous. The worst model only reached an *accuracy* (ratio of correctly classified sentences over total number of sentences) of 47% whereas the best model reached 64% with $n$-grams. This sole metric being unsufficiently relevant to evaluate the classification quality, we considered the *precison* and *recall* which take into account the number of false positives and false negatives. Overall, these results are disappointing inasmuch they are not close to a level that could allow an automatic labelling. They certainly perform better than a random attribution of labels (which gives in average a 14% accuracy) but remain far from a satisfactory classification.

In order to enhance our classification performances, we implemented deep neural networks whose recurrent architectures were perfectly suited to capture language dependencies in both directions (LSTM). We had good hopes to significantly improve our results but it turned out that such models are highly inclined to overfit the and the results were not up to expectations. Indeed, the *accuracy* of the best network was about 60%, that is to say poorer than the best ML model. We eventually tried to use USE pretrained model whose performances were pretty much identical.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 1 | 0.70 | 0.67 | 0.68 | 458 |
| 2 | 0.63 | 0.54 | 0.58 | 252 |
| 3 | 0.51 | 0.32 | 0.39 | 383 |
| 4 | 0.69 | 0.60 | 0.65 | 974 |
| 5 | 0.65 | 0.82 | 0.72 | 1483 |
| 6 | 0.57 | 0.57 | 0.57 | 474 |
| 7 | 0.56 | 0.47 | 0.51 | 310 |
| accuracy | | | 0.64 | 4334 |
| macro avg | 0.62 | 0.57 | 0.59 | 4334 |
| weighted avg | 0.64 | 0.64 | 0.63 | 4334 |

Figure 3: USE classification report

## 4 Discussion/Conclusion

Despite our efforts, we could not reached a classification score that would allow us to automatically label the sense unit. On the one hand, the topics mentionned in a category can be really close to those developed in another one and thus a quite similar vocabulary between categories. Plus, the segmentation of programs into sentences is certainly not suited for such a problem as the context of a sentence plays a lot in its understanding. So getting paragraphs and a redefinition of categories may improve the scores. After that, one might compare different political parties across countries based on their ideologies.

.

## References


Slava Mikhaylov, Michael Laver, and Kenneth R. Benoit. 2011. *"Coder Reliability and Misclassification in the Human Coding of Party Manifestos"*.

Tomas et al. Mikolov. 2013. *"Distributed representations of words and phrases and their compositionality"*. Advances in neural information processing systems.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. *"GloVe: Global Vectors for Word Representation"*. Computer Science Department, Stanford University.

Grégory Piet, Régis Dandoy, and Jeroen Joly. 2015. Comprendre le contenu des programmes électoraux. comparaison des méthodes d'encodage manuel et automatique. *Mots*, 108.

Junbo Zhao Zhang, Xiang and Yann LeCun. 2015. *"Character-level convolutional networks for text classification"*. Advances in neural information processing systems.

Yun Zhu Lilleberg, Joseph and Yanqing Zhang. 2015. *"Support vector machines and word2vec for text classification with semantic features"*. 14th International Conference on Cognitive Informatics Cognitive Computing.

Francois Chaubard, Rohit Mundra, and Richard Socher. 2016. *"Deep Learning for NLP Part-I"*.

Nicolas Merz, Sven Regel, and Jirka Lewandowski. 2016. *"The Manifesto Corpus: A new resource for research on political parties and quantitative text analysis"*.

Dipanjan Sarkar. 2018. *"Sentiment Analysis - Text Classification with Universal Embeddings"*. https://github.com/dipanjanS/data_science_for_all/blob/master/tds_deep_transfer_learning_nlp_classification.

Chris McCormick. 2019. *"BERT Fine-Tuning Tutorial with PyTorch"*. https://mccormickml.com/2019/07/22/BERT-fine-tuning/.