Etude comparative de vecteurs pour l'identification du parti politique d'interventions parlementaires

Pauline Degez and Florian Philippe and Valentine Fleith
Address line
...
43017467@parisnanterre.fr

Abstract

This document is a supplement to the general instructions for *ACL authors. It contains instructions for using the LaTeX style files for ACL conferences. The document itself conforms to its own specifications, and is therefore an example of what your manuscript should look like. These instructions should be used both for papers submitted for review and for final versions of accepted papers.

1 Introduction

La cinquième édition du défi Fouille de Textes (DEFT) porte sur la fouille d'opinions sur des corpus multilingues. Trois tâches ont été proposées, dans trois langues : le français, l'anglais et l'italien. Cet article se concentre sur la 3ème tâche, dont l'objet est l'identification automatique du parti politique d'appartenance de chacun des intervenants dans un corpus de débats parlementaires européens. Il s'agit d'une tâche de classification à 5 classes: `Verts-ALE`, `GUE-NGL`, `PSE`, `ELDR` et `PPE-DE`.

Le but de nos expériences sera ainsi de trouver un/des classifieur(s) permettant de réaliser cette tâche. Pour ce faire, nous utiliserons les algorithmes de Machine Learning implementés dans la bibliothèque Python `scikit-learn`.

1.1 Travaux présentés en 2009

Parti	ELDR	GUE-NGL	PPE-DE	PSE	Verts/ALE
F-mesure	0.21	0.37	0.47	0.37	0.25

Table 1: Moyennes des F-mesures par parti politique.

En 2009, un seul participant a soumis un travail pour la tâche 3 ; la Présentation de l'édition 2009 [1] évoque, pour expliquer cela, les faibles résultats des logiciels sur cette tâche, bien que conformes à ceux que des humains obtiendraient manuellement. L'équipe de l'Université de Montréal (D. Forest and al.) a obtenu en moyenne les f-mesures présentées dans la Table 1. En moyenne, cela donne donc une f-mesure 0.331.

1.2 Notre approche et travaux antérieurs

Pour ce travail, notre approche a été comparative sur plusieurs niveaux. Tout d'abord, nous comparons différents classifieurs : `Random Forest`, `Régression logistique`, `Perceptron` et `Support Vector Machine`. De plus, nous testons aussi différentes vectorisations du corpus sur l'ensemble de ces modèles: `TF-IDF`, `Doc2Vec`, et des `Bert embeddings`.

Plusieurs travaux de recherches explorent les comparaisons entre performances des modèles selon les techniques de vectorisation utilisées. Nous pouvons par exemples évoquer ceux de P. Joseph et S. Y. Yerima [2] en 2022, qui compare les performances des N-grams, TF-IDF, Sac de mots, Word2Vec, Doc2Vec, etc. Leur objectif est de comparer l'impact de la vectorisation sur la précision des modèles. Dans leur article, les modèles `Doc2Vec` et `TF-IDF` démontrent de bons résultats, nous allons ainsi les tester dans notre expérience. Nous décidons d'ajouter à ces deux dernier les embeddings de `BERT` afin d'avoir trois techniques variées : une méthode statistique, une méthode fondée sur un ANN classique et une sur un Transformer.

2 Méthode

2.1 Dataset

Nous utilisons le dataset fournit pour la tâche 3 de l'édition 2009 de DEFT. Au moment de faire des statistiques descriptives, nous nous sommes rendus

[1] Actes du cinquième défi fouille de texte, DEFT2009, Paris, France, 22 juin 2009

[2] P. Joseph and S. Y. Yerima, "A comparative study of word embedding techniques for SMS spam detection," 2022 14th International Conference on Computational Intelligence and Communication Networks (CICN), Al-Khobar, Saudi Arabia, 2022, pp. 149-155,

comptes que le corpus présentait des doublons, et ce majoritairement dans la partition test.

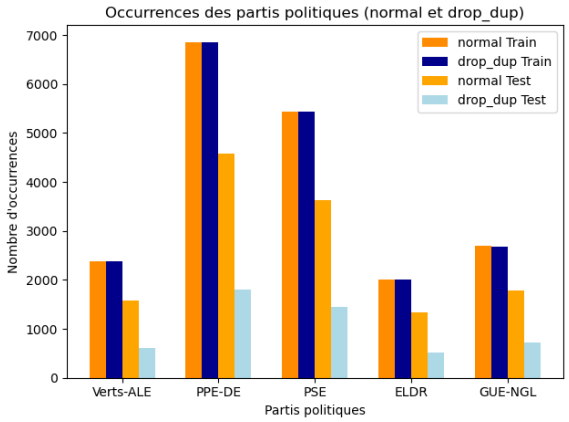

Figure 1: Nombre d'interventions par parti par partition test/train, dans le corpus original et dans la version sans doublons

Après suppression des doublons, la partition prévue (40/60) est changée : elle est maintenant de 20/80[3]. Nous avons envisagé de réimplémenter le partitionnement prévu, mais avons renoncé pour deux raisons : refaire le partitionnement nous éloigne, encore, du corpus initial, et les résultats de quelques modèles sur un corpus repartitionné étaient proches des résultats sur cette partition 20/80. Par ailleurs, la répartition des classes est déséquilibrée : les classes PPE-DE et PSE sont plus grandes et forment à elles deux 63,5 % du corpus. Ceci devra être pris en compte dans le prétraitement.[4]

Statistique	Train	Test
Moyenne	3871.4	1021.2
STD	2149.1	569.7
Min	2005.0	525.0
1er quartile	2376.0	615.0
Médiane	2687.0	715.0
3eme quartile	5431.0	1448.0
Max	6858.0	1803.0

Table 2: Nombre d'intervention des partis par partition

2.2 Prétraitements

Le texte des interventions a été soumis à un prétraitement simple :

(1) Suppresion de la ponctuation

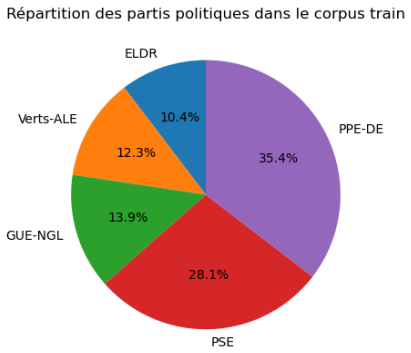

Figure 2: Répartition des interventions par parti dans la partition train sans doublons

(2) Unification de la casse en minuscules

(3) Tokenisation[5]

Pour résoudre le problème de déséquilibre des classes, nous avons opté pour le *downsampling* afin d'obtenir des classes relativement équilibrées, en utilisant la fonction *resample* de la bibliothèque `sckit-learn`. Après *downsampling*, les classes PPE-DE et PSE ne représentent plus que 43% du corpus, ayant chacune été ramenée autour de 21,5% du corpus

Parti	Train	Test
ELDR	2531 (16,1%)	525 (16,2%)
PPE-DE	3402 (21,63%)	715 (22%)
GUE-NGL	3402 (21,63%)	687 (21,2%)
PSE	3402 (21,63%)	693 (21,4%)
Verts-ALE	2990 (19%)	614 (19%)
Total	15727	3235

Table 3: Nombre d'intervention par parti par partition pour une langue après *downsampling*, example de l'italien

2.3 Les différents embeddings testés

Nous avons choisi dans notre étude de comparer les résultats obtenus sur une tâche de classification en utilisant 3 techniques de vectorisation différentes.

La vectorisation `TF-IDF`[6](*Term Frequency-Inverse Document Frequency*) est la méthode la plus "ancienne" que nous présentons. Elle

[3] 0.79 pour le train et 0.21 pour le test

[4] la figure correspond au train, mais la répartition est sensiblement la même dans le test

[5] Une lemmatisation avec la bibliothèque `SpaCy` a été envisagée, mais ce corpus multilingue aurait nécessité le chargement de 3 modèles linguistiques différents et ralongé d'autant le temps de traitement

[6] Implémentée avec la fonction `tfidfvectorizer` de `scikit-learn`

se base sur une mécanique de comptage des mots: la fréquence d'apparition de chaque mot dans un document est divisée par sa fréquence d'apparition dans le corpus, permettant de donner plus d'importance aux mots significatifs dans leurs documents d'apparition.

La vectorisation `Doc2Vec`[7] qui génère des vecteurs de de document plutôt que de mot. Ces vecteurs sont l'ouput d'un réseau de neurones et nécessitent donc une phrase d'entrainement. Pour cette étude, nous avons choisi de générer des vecteur Doc2Vec à 100 dimensions[8] avec une fenêtre glissante de 5 mots et d'ignorer les mots n'apparaissant pas au moins 3 fois.

La vectorisation avec `BERT` multilingue[9] qui se base sur les réseaux de neurones mais s'appuie sur une architecture transformer et nécessite également une phase d'entrainement. Contrairement à Doc2Vec, elle renvoie des vecteurs de mots, et utilise un mécanisme d'attention lors de la génération des vecteurs, lui permettant de prendre en compte l'importance d'un mot en fonction du contexte local dans lequel in apparait. Nous avons généré des vecteurs à 768 dimensions (valeur de base).

3 Résultats

Comme mentionné en introduction, nous avons testé trois méthodes de vectorisation : TF-IDF, doc2vec et BERT Nous avons utilisé ces vecteurs pour entraîner et comparer 4 modèles différents : Régression logistique, SVM, Random Forest et Perceptron.

3.1 Vecteurs BERT

La figure 3 montre qu'avec les vecteurs BERT, la régression logistique obtient les meilleurs résultats dans les trois langues.

3.2 Vecteurs doc2vec

Comme on peut le voir sur la figure 4, avec les vecteurs doc2vec, c'est le SVM qui obtient les meilleurs résultats dans les trois langues.

3.3 Vecteurs TF-IDF

Les meilleurs résultats que nous ayons obtenus sont ceux avec un vectorisation tf-idf. On peut voir sur la figure 5 que nous avons obtenu au mieux :

[7] Implémentée à l'aide de la bibliothèque `gensim`

[8] il serait possible de faire plus, mais nous essayions de ne pas saturer nos machines

[9] bert-base-multilingual-uncased

Figure 3: Résultats obtenus avec les vecteurs BERT.

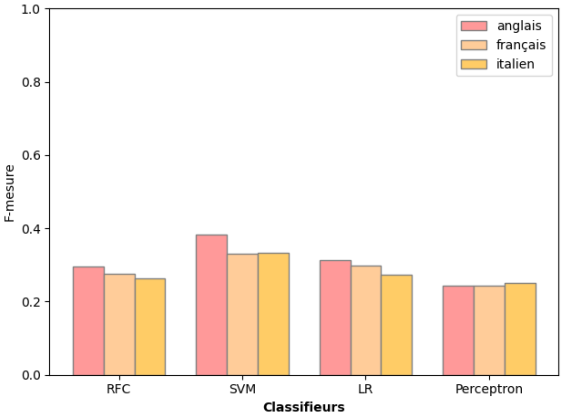

Figure 4: Résultats obtenus avec les vecteurs doc2vec.

0.46 en anglais avec un SVM, 0.43 en italien avec une régression linéaire et 0.44 en français avec un SVM.

4 Preamble

The first line of the file must be

```
\documentclass[11pt]{article}
```

To load the style file in the review version:

```
\usepackage[review]{acl}
```

For the final version, omit the `review` option:

```
\usepackage{acl}
```

To use Times Roman, put the following in the preamble:

```
\usepackage{times}
```

Please see the LaTeX source of this document for comments on other packages that may be useful.

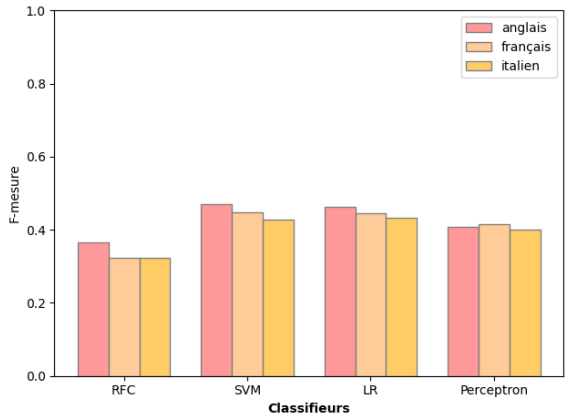

Figure 5: Résultats obtenus avec les vecteurs TF-IDF.

Command	Output	Command	Output
{\"a}	ä	{\c c}	ç
{\^e}	ê	{\u g}	ğ
{\`i}	ì	{\l}	ł
{\.I}	İ	{\~n}	ñ
{\o}	ø	{\H o}	ő
{\'u}	ú	{\v r}	ř
{\aa}	å	{\ss}	ß

Table 4: Example commands for accented characters, to be used in, *e.g.*, BibTEX entries.

By default, the box containing the title and author names is set to the minimum of 5 cm. If you need more space, include the following in the preamble:

```
\setlength\titlebox{<dim>}
```

where `<dim>` is replaced with a length. Do not set this length smaller than 5 cm.

5 Document Body

5.1 Footnotes

5.2 Tables and figures

See Table 3 for an example of a table and its caption.
Do not override the default caption sizes.

As much as possible, fonts in figures should conform to the document fonts. See Figure 6 for an example of a figure and its caption.

environment at an appropriate point within the text. The `graphicx` package supports various optional arguments to control the appearance of the figure. You must include it explicitly in the LATEX preamble (after the `\documentclass` declaration and before `\begin{document}`) using `\usepackage{graphicx}`.

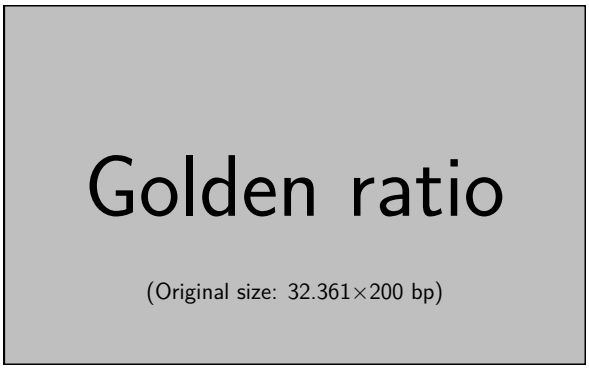

Figure 6: A figure with a caption that runs for more than one line. Example image is usually available through the `mwe` package without even mentioning it in the preamble.

5.3 Hyperlinks

Users of older versions of LATEX may encounter the following error during compilation:

This happens when pdfLATEX is used and a citation splits across a page boundary. The best way to fix this is to upgrade LATEX to 2018-12-01 or later.

5.4 Citations

Table **??** shows the syntax supported by the style files. We encourage you to use the natbib styles. You can use the command `\citet` (cite in text) to get "author (year)" citations, like this citation to a paper by Gusfield (1997). You can use the command `\citep` (cite in parentheses) to get "(author, year)" citations (Gusfield, 1997). You can use the command `\citealp` (alternative cite without parentheses) to get "author, year" citations, which is useful for using citations within parentheses (e.g. Gusfield, 1997).

A possessive citation can be made with the command `\citeposs`. This is not a standard natbib command, so it is generally not compatible with other style files.

5.5 References

The LATEX and BibTEX style files provided roughly follow the American Psychological Association format. If your own bib file is named `custom.bib`, then placing the following before any appendices in your LATEX file will generate the references section for you:

```
\bibliography{custom}
```

You can obtain the complete ACL Anthology as a BibTEX file from https://aclweb.org/

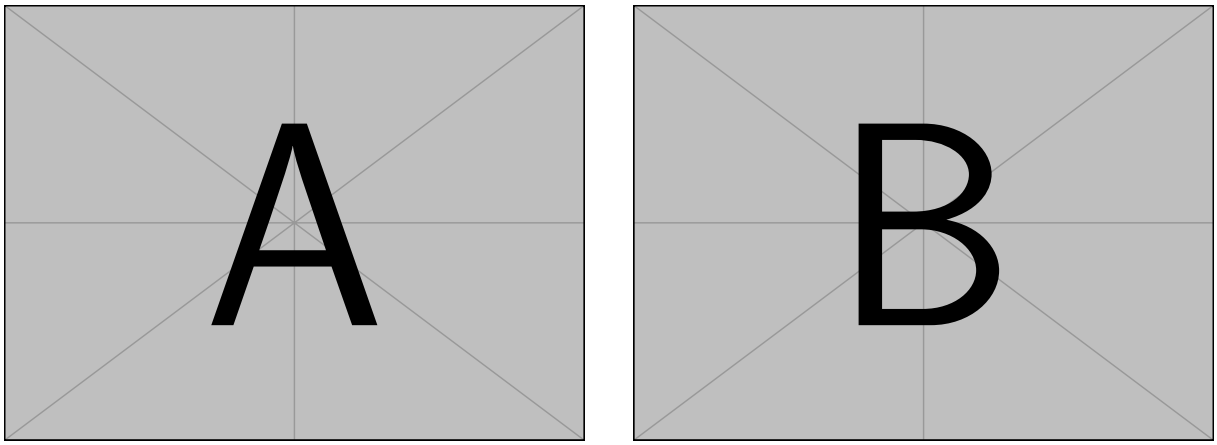

Figure 7: A minimal working example to demonstrate how to place two images side-by-side.

anthology/anthology.bib.gz. To include both the Anthology and your own .bib file, use the following instead of the above.

```
\bibliography{anthology,custom}
```

Please see Section 6 for information on preparing BibTEX files.

5.6 Equations

An example equation is shown below:

$$A = \pi r^2 \tag{1}$$

Labels for equation numbers, sections, subsections, figures and tables are all defined with the `\label{label}` command and cross references to them are made with the `\ref{label}` command.

This an example cross-reference to Equation 1.

References

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853.

Galen Andrew and Jianfeng Gao. 2007. Scalable training of L1-regularized log-linear models. In *Proceedings of the 24th International Conference on Machine Learning*, pages 33–40.

Dan Gusfield. 1997. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, Cambridge, UK.

Mohammad Sadegh Rasooli and Joel R. Tetreault. 2015. Yara parser: A fast and accurate dependency parser. *Computing Research Repository*, arXiv:1503.06733. Version 2.

5.7 Appendices

Use `\appendix` before any appendix section to switch the section numbering over to letters. See Appendix A for an example.

6 BibTEX Files

Unicode cannot be used in BibTEX entries, and some ways of typing special characters can disrupt BibTEX's alphabetization. The recommended way of typing special characters is shown in Table 3.

Please ensure that BibTEX records contain DOIs or URLs when possible, and for all the ACL materials that you reference. Use the `doi` field for DOIs and the `url` field for URLs. If a BibTEX entry has a URL or DOI field, the paper title in the references section will appear as a hyperlink to the paper, using the hyperref LATEX package.

Acknowledgments

This document has been adapted by Steven Bethard, Ryan Cotterell and Rui Yan from the instructions for earlier ACL and NAACL proceedings, including those for ACL 2019 by Douwe Kiela and Ivan Vulić, NAACL 2019 by Stephanie Lukin and Alla Roskovskaya, ACL 2018 by Shay Cohen, Kevin Gimpel, and Wei Lu, NAACL 2018 by Margaret Mitchell and Stephanie Lukin, BibTEX suggestions for (NA)ACL 2017/2018 from Jason Eisner, ACL 2017 by Dan Gildea and Min-Yen Kan, NAACL 2017 by Margaret Mitchell, ACL 2012 by Maggie Li and Michael White, ACL 2010 by Jing-Shin Chang and Philipp Koehn, ACL 2008 by Johanna D. Moore, Simone Teufel, James Allan, and Sadaoki Furui, ACL 2005 by Hwee Tou Ng and Kemal Oflazer, ACL 2002 by Eugene Charniak and Dekang Lin, and earlier ACL and EACL formats

written by several people, including John Chen, Henry S. Thompson and Donald Walker. Additional elements were taken from the formatting instructions of the *International Joint Conference on Artificial Intelligence* and the *Conference on Computer Vision and Pattern Recognition*.

References

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853.

Galen Andrew and Jianfeng Gao. 2007. Scalable training of L1-regularized log-linear models. In *Proceedings of the 24th International Conference on Machine Learning*, pages 33–40.

Dan Gusfield. 1997. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, Cambridge, UK.

Mohammad Sadegh Rasooli and Joel R. Tetreault. 2015. Yara parser: A fast and accurate dependency parser. *Computing Research Repository*, arXiv:1503.06733. Version 2.

A Example Appendix

This is an appendix.