Analyse de la parole politique avec LDA

Amal Targhi Katia Sana Mohamed Mohamed Maxime Buron
Mahmut Cavdar

February 10, 2017

1 Introduction

Si l'on voulait résumer le métier de journaliste en un verbe, ça serait le verbe choisir. Des normes quantités d'informations qui paraissent chaque jour dans les médias à travers le monde, pour préseneter un contenu attactif le journaliste doit réaliser un travail d'investigation autour des thématiques d'actualité, qui sucitent beacuoup le réactions. l'objectif de ce projet est de réaliser un systeme permettant aux journalistes de selectionner les sujets politiques qui vont leur interesser (par ordre) à partir d'une base de données de tweets des personnages politiques en appliquant un topic model de type LDA.

2 Latent Dirichlet allocation

2.1 Définition

Latent Dirichlet allocation (LDA) est un modèle probabiliste [2] qui permet d'expliquer des ensembles d'observations par des variable non observables (latente) où les mots sont les variables observables et les sujets sont les variables latentes. Cet algorithme a pour but de découvrir automatiquement les thèmes d'un corpus. Par exemple:
Supoosant que nous avons les 5 phrases suivant:

- j'aime du broccoli et du banane.
- je mange de la banane et du poulet pour le petit déjeuner.
- les chats et les chiens sont mignons.
- ma soeur a adopté un chat hier.
- regarde ce hamster qui croque le broccoli

Qu'est-ce que l'allocation latente de Dirichlet? C'est une façon de découvrir automatiquement les sujets contenus dans ces phrases. Par exemple, étant donné ces phrases et nous avons besoins de 2 sujets, LDA pourrait produire quelque chose comme suit:

- Phrase 1 et 2: 100% topic A.
- Phrase 3 et 4: 100% topic B.
- Phrase 5: 60% Topic A, 40% Topic B
- Topic A: 30% broccoli, 15% bananas, 10% petit dejeuner, 10% croquer... (on peut interpreter que topic A est nouriture).
- Topic B: 20% chien, 30%chat, 10% mignions, 15% hamster... On peut interpreter que topic B est animal.

La question qu'on peut poser comment LDA marche?

2.2 Modèle

Latent Dirichlet allocation (LDA) modèle probabiliste générative. L'idée de base est: les documents sont représentés comme des mélanges aléatoires (random mixture) des sujets(latente) où chaque sujet est caractérisé par une distribution de mots. LDA suit le processus génératif suivant pour chaque document dans un corpus:

1. Décidez du nombre de mots N dans un document (selon une distribution de Poisson)
2. Attribuer un thème pour chaque document (selon la distrubution de dirichlet pour un ensemble de sujets K (constante)).
3. Genérer chaque mot w_i dans le document D par :
   (a) Attribuer un thème selon la distribution multinomiale (les thèmes attribués par la distribution de dirichlet 2ème étape)
   (b) Utiliser le topic pour générer le mot lui-même (selon la distribution multinomiale du sujet). Par exemple, si nous avons choisi le sujet nourriture, nous pourrions générer le mot «brocoli» avec 30% de probabilité, «bananes» avec une probabilité de 15%, et ainsi de suite.

2.3 Apprentissage

Supposnat que nous disposons d'un ensemble de documents, nous avons choisi un nombre fixe K de sujets qu'on veut découvrir et nous voulons utiliser le modèle LDA afin d'apprendre la représentation des thèmes de chaque document et les mots associés à chaque document. On aboutira à ça suivant collapsed Gibbs sampling comme suit:

1. Parcourir chaque document et assigner au hasard chaque mot du document à l'un des K sujets (alétoirement)
2. On doit améliorer ces assignement car la première initialisation est aléatoire. Pour ceci: pour chaque document d:
   - pour chaque word w dans d
     (a) Pour chaque topic t, calculer deux choses: 1) P(topic t| document d) et 2) P(mot w | topic t) . Réaffecter un nouveau le topic, où nous choisissons le topic t avec la probabilité p (topic t | document d) * p (mot w | sujet t) (selon notre génératif modèle, c'est essentiellement la probabilité que le topic t génère le mot w, Il est logique que nous rééchantillonnions le sujet du mot actuel avec cette probabilité).
     (b) En d'autres termes, dans cette étape, nous supposons que toutes les assignations de sujets à l'exception du mot en question sont correctes, puis en mettant à jour l'affectation du mot courant en utilisant notre modèle de génération des documents.
3. Après avoir répété l'étape précédente un grand nombre de fois, on finit par atteindre un état approximativement stable où les affectations sont assez bonnes. Ces affectations sont utilisées pour estimer les topic mixtures de chaque document.

3 Présentation et Préparation des données pour le LDA

3.1 Présentation des données

Pour mettre en œuvre ce projet, nous disposons des fichiers csv contenant les tweets politiques (des politiciens etc ...) ainsi qu'un fichier contenant "les users names des parties politiques et leurs catégorie". Ces tweets sont datés de janvier 2015 à décembre 2016.
Le module LDA fonctionne mal dans des environnements de textes courts comme publication Facebook, Twitter etc... Donc pour résoudre ce problème il faudra agréguer les tweets afin de créer des documents virtuels plus larges sur les quelles on poura appliquer notre topic model.

3.2 Agrégation

Avec la croissance explosive des réseaux sociaux tels que Twitter et Facebook, les textes courts sont devenus le format prédominant pour les informations issus d'internet [7]. Par exemple, environ 250 millions d'utilisateurs actifs sur Twitter peuvent générer près de 500 Millions de tweets par jour. Ce volume énorme de textes courts contient des informations sophistiquées qui ne peuvent être trouvées dans les sources d'information traditionnelles. Les topic modèles probabilistes ont été largement utilisés pour extraire automatiquement les thèmes d'un grand corpus. Les topic model standards [4] [2] supposent qu'un document est généré à partir d'un mélange de topics, où un topic est une distribution probabiliste des mots. LDA est l'un des topic models qui ont connu un grand succès cepedant les résultat étaient mauvais quand le model est appliqué directement sur des document de petite taille tel que tweets [5], car dans LDA un document est considéré comme un mélange des topics et un mot est tiré de l'un de ces topics. Et si nous avons un court document avec seulement quelques mots les observations sont évidemment trop peu nombreuses pour déduire les résultats. L'agrégation est une stratégie simple, largement adoptée pour les textes courts, elle consiste à regrouper des textes courts en des plus larges avant d'appliquer LDA. Par exemple, les tweets contiennent non seulement du contenu textuel mais aussi des informations contextuelles telle que les hashtags, les dates et les noms des utilisateurs. Ces informations contextuelles peuvent être utilisé pour agréger les tweets avant d'effectuer la modélisation des thématiques.
Dans ce projet nous avons opté pour effectuer trois méthodes d'agregation:

Agregation par Hashtag:

L'ensemble des document contient 3674326 hashtags où un hashtag est répété dans plusieurs tweets. Après avoir agregé les tweets par hashtag nous avons obtenu 111925 documents. Nous avons pensé à supprimer les documents avec peu de tweets, en effet nous avons fixé un seuil de 3 càd les document contenant moins de 3 tweets seront effacé et nous avons réduit le nombre de document de 111925 à 26 114 mais ceci n'a pas donné de bons résultats également nous avons fixé un seuil de 2 c'est à dire supprimer les tweets avec moins de 2 tweets, et les topics retourné n'étaient pas pertinents, d'après la figure 1 on peut remarquer que nous avons 60641 document avec un seul tweet, 2 documents avec 17145 tweets, 3 documents avec 8025 tweets,4 document avec 4678 tweets et 5 documents avec 3038 tweets ... Nous avons également remarqué qu'il y a beaucoup de documents qui contiennent des tweets repetés celà est dû au retweets, nous avons élminié les tweets repetés en gardant juste une seul occurence en utilisant le module md5 de python cependant les résultats n'étaient pas très pertinents.

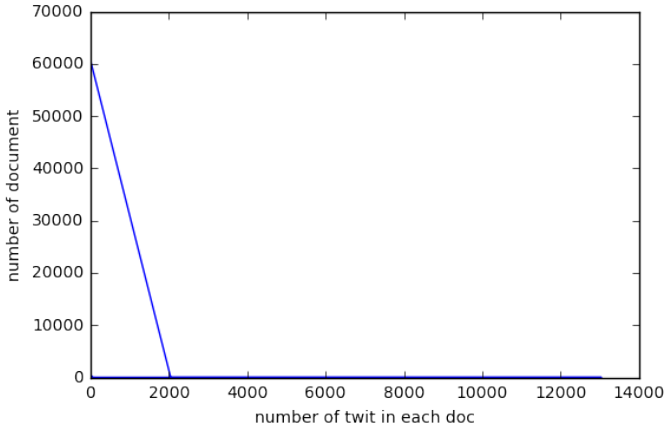

Figure 1: Le nombre des document par rapport au nombre des tweets dans chaque document

Agregation par catégorie politique et date:
Dans cette partie nous avons regroupé les tweets par catégorie politique (8 catégories) et par une plage de date (échantillonage d'une semaine) par exemple : pour la catégorie droite, chaque document va contenir les tweets de la droite dans chaque semaine. En global nous avons eu un peu près de 800 documents.

Agregation par utilisateur:

Cette méthode consiste a agréger les tweets par utilisateur, mais elle n'aboutie pas à un bon résultat par ce que chaque utilisateur aborde plusieurs thématiques différentes pour cela nous ne l'avons pas utilisée.

3.3 Pre-processing

Dans toute application avec un jeu de données textuelles volumineux, il est primordial de traiter les données afin de les adapter au mieux à l'usage qu'on veut faire. La complexité et la nature des ces pré-traitement dépent en premier lieu du langage utilisé. Dans notre cas puisque le français est le langage utilisé, voici la liste des pré-traitement effectués.

1. Tokenisation: Étant donnée une séquence de mots dans un document, la tokenisation est la tâche de découpage en mots, appelés tokens. (module TweetTokenizer de NLTK).
2. transformer tous les caractères majuscules en caractères minuscules.
3. Suppression de la ponctuation, les accents et les mot contenant moins de 2 caractères par exemple les caractères, l'alphabet etc...
4. Suppression des mots vides (stopwords), sont des mots qui sont tellement communs qu'il est inutile de les indexer ou de les utiliser dans une recherche. En français, des mots vides évidents pourraient être « le », « la », « de », « du », « ce »...
5. Suppression des emoji, des symboles,flags etc ...

Le coprus pré-traité produit à l'issu de toutes les étapes décrites ci-dessous est donné en entrée à l'algorithme LDA qui va en extraire les différents topics pour chaque document. Nous avons choisi d'utiliser l'implémentation de LDA présenté dans la librairie Gensim. Gensim est une librairie open source de Python dediée au traitement de grandes quantités de texte. Elle fournit les outils nécessaires principalement au topic modeling et modèle vectoriel.

4 Visualisation

En visualisant les résultats du LDA, on cherche à comprendre l'interprétation de chaque topic (1). Pour comprendre un modèle en entier, il ne suffit pas de se focaliser sur un topic mais plutôt l'ensemble des topics, cela pour découvrir comment les topics sont liés entre eux (2) [6].

4.1 Cloud world

On poura interpreter un thème en l'explorant à travers les mots qui le composent ainsi que leurs probalbilité. Pour cela, on crée un worldcloud qui est tout simplement une représentation graphique des fréquences de mot. Pour ceci nous avons utilisé la libraire wordcloud de python.

Figure 2: Le word cloud

4.2 PyLDAvis

LDAvis est une librairie de vizualisation python qui permet aux utilisateurs d'interpreter leurs topics generés par LDA. LDA propose :

- Une meilleure interprétation des topics en affichant ses mots les plus pertinents.
- Visualiser la pertinence des topics: dans LDAvis, les topics sont représentés par des cercles, la taille de chacun d'eux reflète la pertinence du topic par rapport au corpus.

4.3 JavaScript

Pour résumer en utilisant ces méthodes de visualisation (wordcloud, pyLDavis) nous avons pu visualisé les résultats généré par LDA et répondre aux question que nous avons posé au début (Interpreter chaque topic ainsi que la relation entre eux).

Nous avons étendu la librairie de LDAvis qui permet l'affichage des topics et des mots qu'ils contiennent. Cette extension concerne deux options qui n'était pas fourni par l'interface mais qui sont en pratique pour le journaliste :

- renommer les thèmes
- fusionner différents thèmes

Les données nécessaires à LDAviz sont stockées dans un fichier lda.json. Nous avons décidé d'ajouter un ficher de configuration mapping.json qui est utilisé par notre extension et contient les noms et les fusions à effectuer. Notre extension effectue uniquement un pré-traitement sur les données à visualiser, elle charge lda.json effectue les fussions directement sur les données brutes et excute l'affichage de LDAviz sur les nouvelles données.

5 Mise en oeuvre d'un moteur de recherche

pour réaliser cette partie nous avons crée un data frame qui contient tous les probabilités possibles : $P(date, catgorie, thme) = P(thme|date, catgorie) \div \sum P(date, catgorie, thme')$

tel que le (théme)' c'est la block sélectionné par la requête entrer.
la probabilité P(théme|date , catégorie) est récupirer à partir de topic model LDA.

1. Requête (date,catégorie)
ce graphe montre la distrbution probabiliste des thémes par apport a la requête enter:

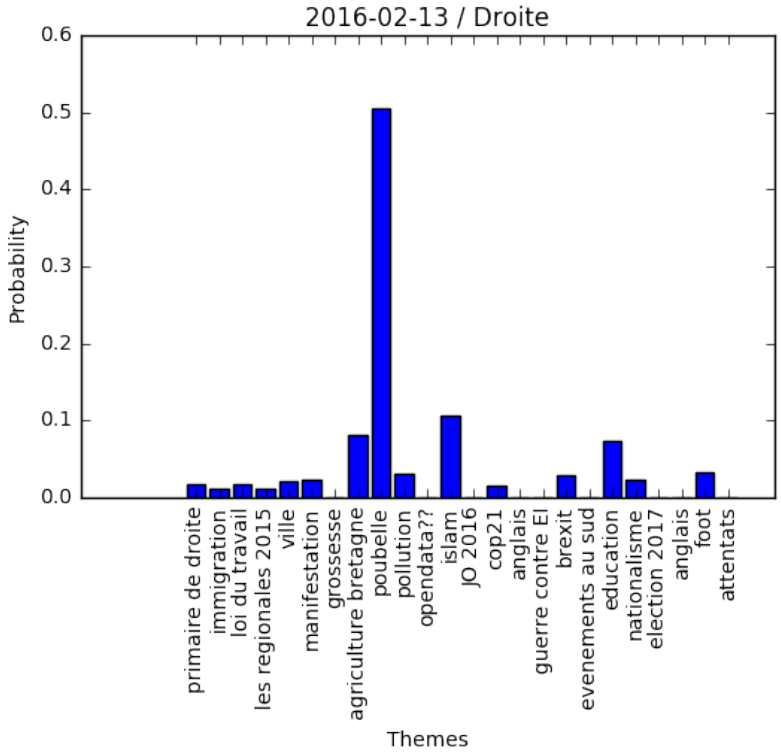

Figure 3: Résultats avec la requête : (date,catégorie) ==> thème

2. Requête (catégorie,théme)
ce graphe montre la distrbution probabiliste des dates par apport a la requête enter:

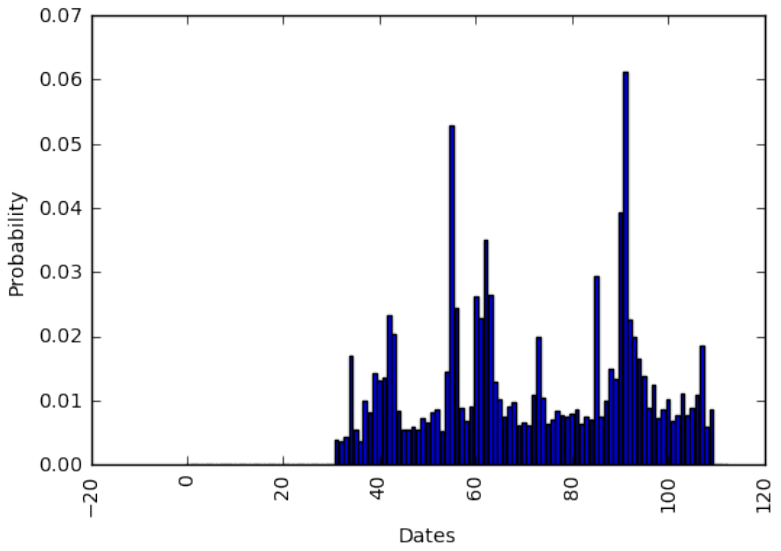

Figure 4: Résultats avec la requête : (catégorie,théme) ==> Date

3. Requête (date,théme)

ce graphe montre la distrbution probabiliste des catégories par apport a la requête enter:

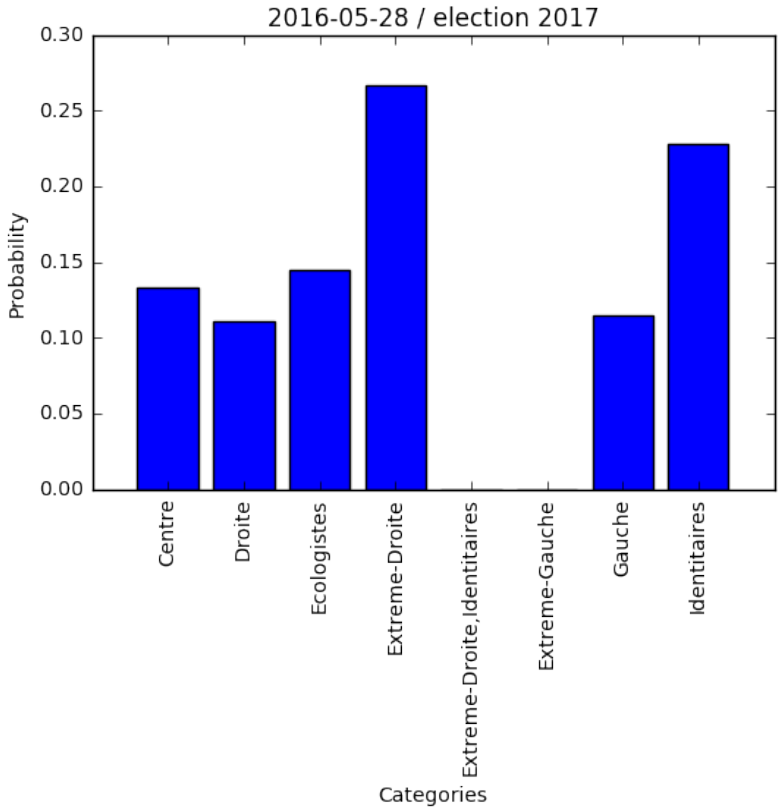

Figure 5: Résultats avec la requête : (date,théme) ==> catégorie

6 Tests et résultats

6.1 Evaluation du LDA

Le Topic Modeling apprend de manière non supervisée sur un les topics, représentées par un ensemble de documents non étiquetées. Les topics appris par LDA sont utiles mais leur interpretation reste diffile. Un des problèmes de ce topic modèle et les topic poubelle "junk topic" en effet, on poura trouver des topic avec des mots incohérant non signifcatif par exemple: un topic qui contient les mots suivant " vacances, attentas, noixdecoco, cuisine, marrakech, vivelemaroc, brexit, cop21, bmw, tanger ...). Les résultat doivent être souvent analysé par des experts humains afin de valider un thème (poubelle ou non). Dans la littérature plusieurs travaux ont essayé de simuler le jugement humain afin de valider un topic en se basant sur des mesures probabilistes [1][3]. Cependant dans notre cas, c'est nous qui avons évalué les topics retournés par le modèle, nous avons detecté 13 topics poubelles et 27 topics pertinentss (Bien cohérents) et il y avait des topics qui se ressemblent, de ce fait nous les avons fusionnés et nous avons enfin eu 25 topics que nous avons renommé nous-même.

6.2 Choix du nombre de topics

Le LDA est une technique non supervisée pour la découverte de topics dans un corpus. Le nombre de topics est donné en entrée et il est fixe, le choix de ce ce paramètre est une tâche qui s'avère difficle. On ne peut pas l'estimer de manière précise. Dans notre cas on a essayé pour plusieurs valeurs qui varirent entre 20 et 50. et le meilleur résultat est 40 topics.

Nombre de topics	Résultat
20	rousset 2015, bruxelle, bordeaux, france, erasmus, debatpso , sncf, apprentissage, limoges ... Fillon, Sarkozy, Dan, France plu, debat, pamaredoite, vou, mai ...
30	jeunes, entreprise, paris, soir, emploi, etrepreneurs, formation, revue, region, recherche, maire edf, nucelaire,centrale, brexit, streetart, eelv, ue, europe, français, avenir, projet
40	cop21, paris, climat, conseilledeparis, cop22, climatique, pollution ... terrorisme, france, attentats, saint-denis, victimes, terrorsites, bruxelles, hommage, paris, fusillade ...
50	cinema, photo, vtc, patriosphere, livre, taxis, culture, festival, france, edition , merci, recherche ... immigration, cologne, clandestins, migrants, migrant, acceuil, callais, refugies, allemagne, jerusalem...

6.3 Résulats de topic modeling LDA

Nous avons utilisé LDA pour extraire les topics des tweets. Bien que les résulats obtenues sont intéressants, ces résultats pourraient être amélioré car d'une part nous avons beacoup de poubelle voir figure 6 d'autre part on peut avoir une sorte d'impurté dans les topic par exemple dans le topic cop21 par contre le modèle a detecte le mot tafta en tant que mot important même si les autres mots sont cohérants, ce qui peut perturber provoquer une sorte de pertubation voir figure 8. notre LDA detecte les mots étrangers, en effet le modéle apprends les mots anglaise. ce qui explique la performance de note modèle, voir figure 7 (topic anglais). Les résultats de note modèles sont pertinents, les topic sont cohérents ceci est exprimé par la similarité sémantique entre les mots de chaque topic. A titre d'exemple les topic agriculture bretagne et primaire droite etc... voir figure 10 9.

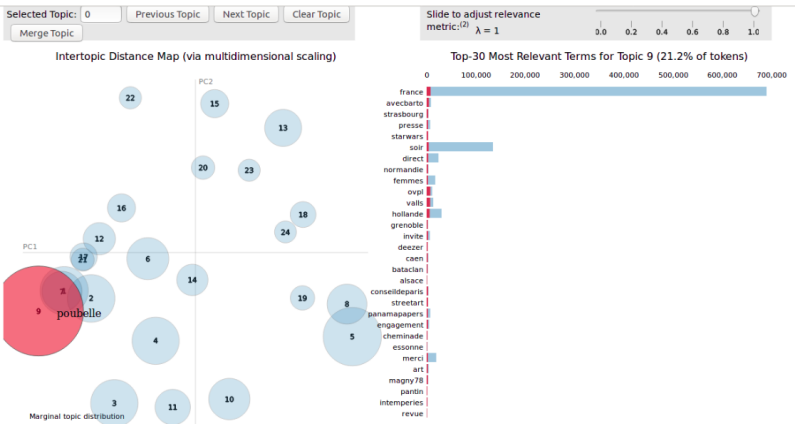

Figure 6: Résultats avec LDAvis : topic poubelle

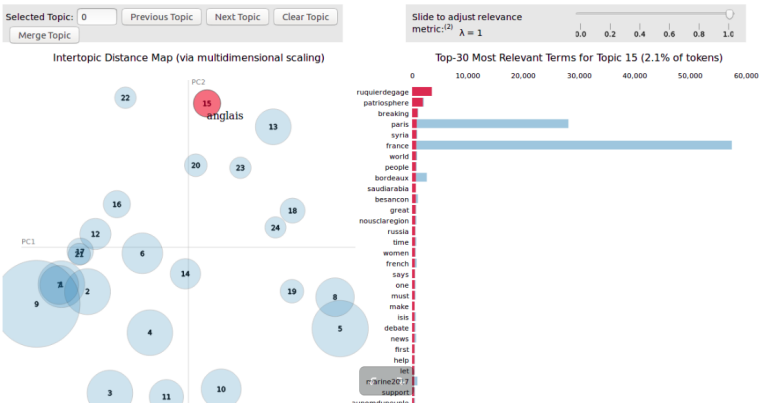

Figure 7: Résultats avec LDAvis : topic anglais

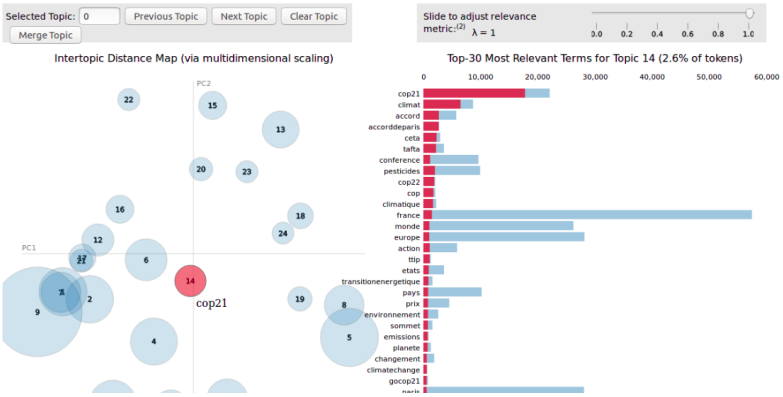

Figure 8: Résultats avec LDAvis : topic cop21

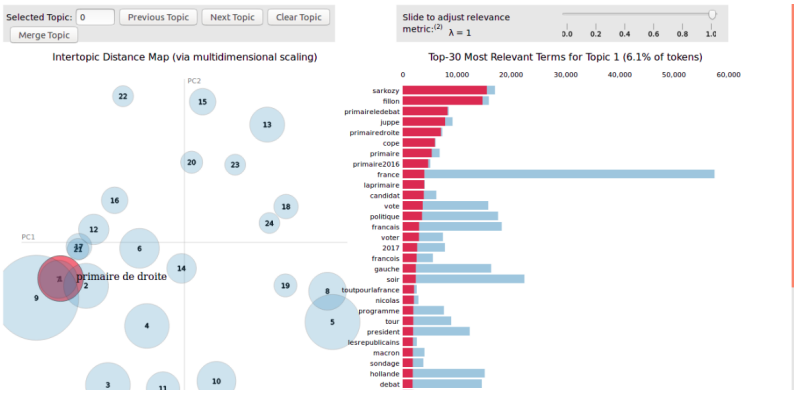

Figure 9: Résultats avec LDAvis : topic primaire droite

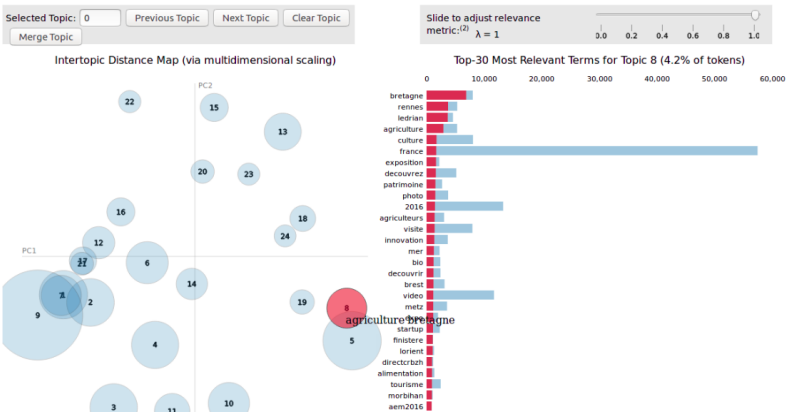

Figure 10: Résultats avec LDAvis : topic agriculture en bretagne

7 Conclusion

Pour conclure, on peut dire que nous avons réussi à faire du Topic Modeling sur notre corpus, trouver différentes manières de visualisations des topics afin de tirer plus d'informations. En effet nous avons agregé le tweets, appliqué le topic modèle sur la collection obtenue, réaliser une interface de visualisation des résultat et mettre un moteur de recherche avec les topics choisis. Il s'avère également que le pré-traitement des données effectué est cruciale, en effet nous avons eu des topics pertinents. Des améliorations peuvent être apporté à notre travail tel que: l'utilisation des thèmes choisis pour quelques études.

Contents

References

[1] Loulwah AlSumait, Daniel Barbará, James Gentle, and Carlotta Domeniconi. Topic significance ranking of lda generative models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 67–82. Springer, 2009.

[2] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[3] Gerlof Bouma. Normalized (pointwise) mutual information in collocation extraction. In *Proceedings of the Biennial GSCL Conference*, volume 156, 2009.

[4] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

[5] Liangjie Hong and Brian D Davison. Empirical study of topic modeling in twitter. In *Proceedings of the first workshop on social media analytics*, pages 80–88. ACM, 2010.

[6] Carson Sievert and Kenneth E Shirley. Ldavis: A method for visualizing and interpreting topics. In *Proceedings of the workshop on interactive language learning, visualization, and interfaces*, pages 63–70, 2014.

[7] Yuan Zuo, Junjie Wu, Hui Zhang, Hao Lin, Fei Wang, Ke Xu, and Hui Xiong. Topic modeling of short texts: A pseudo-document view. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2105–2114. ACM, 2016.