Institut National des Langues et Civilisations Orientales

Département Textes, Informatique, Multilinguisme

Analyse de sentiments pour mesurer la perception d'ArcelorMittal sur divers sujets clés

MASTER

TRAITEMENT AUTOMATIQUE DES LANGUES

Parcours :

Ingénierie Multilingue

par

Shéhérazade NINEB

Directeur de mémoire : Patrick Paroubek

Encadrants :

Nicolas Bontems
Diego Diaz Fidalgo

Année universitaire 2020/2021

TABLE DES MATIÈRES

Liste des figures

Liste des tableaux

Remerciements

Je tiens à remercier l'ensemble des enseignants du Master TAL de l'INALCO pour la richesse de leurs enseignements et leur bienveillance.

Je remercie Madame Kata Gabor pour m'avoir fait découvrir le TAL en LLCE.

Je remercie ArcelorMittal pour ce stage passionnant qui m'a fait renouer avec la recherche.

Je remercie Nicolas Bontems pour son dévouement, son énergie, son soutien.

Je remercie les membres de l'équipe qui ont soutenus ce projet. Je les remercie pour leurs conseils, pour leur temps en particulier Diego Diaz Fidalgo, Jorge Rodil Martinez et Kris Kupilas.

J'ai une pensée pour mon équipe que je remercie pour son ouverture en particulier Gérard Pasquet et Xavier Segas.

Enfin je remercie particulièrement mon directeur de mémoire, Patrick Paroubeck, pour ses précieux conseils, sa patience, sa gentillesse et tout ce temps qu'il a pu m'accorder.

RÉSUMÉ

La réputation des entreprises impacte directement leur pouvoir d'attraction, leur capacité à attirer des talents mais aussi leur croissance. Elles se soucient donc de leur identité en ligne, cherchent à mieux connaître les attentes et critiques que les internautes leur adressent. L'analyse de sentiments met en œuvre des techniques de calculs du Traitement Automatique du Langage Naturel qui répondent à cette problématique. Les travaux de ce mémoire consistent à d'une part fournir une vue d'ensemble des méthodes et outils existants, et d'autre part à démontrer la capacité de ces méthodes à analyser la réputation d'ArcelorMittal sur divers sujets. La première partie de ces travaux à consister en une analyse du sentiment à l'échelle du document. La seconde partie à l'analyse du sentiment à l'echelle des aspects. Des approches de types lexicon-based, Deep Learning (BERT) et de Topic Modeling ont été mis en œuvre.

INTRODUCTION

L'analyse de sentiments connait un essor au début des années 2000 depuis l'entrée en scène du Web 2.0 participatif et de l'utilisation massive du Big Data et des réseaux sociaux. L'intérêt va croissant pour connaître les opinions d'internautes qui s'expriment spontanément et en temps réel. Ces masses de données accessibles constituent de véritables gisements exploitables.*« Ce que pensent et disent les gens »* est régulièrement convoqué pour la prise de décision que ce soit en vue d'un achat, d'une réservation d'hôtel, d'un contexte électoral, d'un suivi du marché financier ou encore pour évaluer la réputation de son entreprise. Les marques se soucient de leur identité en ligne, cherchent à mieux connaître les attentes et critiques que les internautes leur adressent [BOULLIER and LOHARD, 2012]. La réputation des entreprises impacte directement leur pouvoir d'attraction, leur capacité à attirer des talents mais aussi leur croissance.

L'analyse de sentiments, aussi connue sous le nom de fouille d'opinions, fait référence aux techniques de calculs mise en œuvre en Traitement Automatique du Langage Naturel visant à classifier, qualifier des opinions exprimées par des locuteurs à des échelles plus ou moins précises.

ArcelorMittal R&D a lancé fin 2018 un projet de transformation visant à adopter des principes d'organisation et de travail caractérisant l'organisation exponentielle. L'objectif final est clair : faire partie des 50 entreprises les plus innovantes. Les méthodes de traitement du langage naturel ont été, entre autres, identifiées comme pouvant aider à mieux mesurer la perception d'ArcelorMittal et à identifier des domaines d'amélioration.

Les travaux de ce mémoire consistent à d'une part fournir une vue d'ensemble des méthodes et outils existants, et d'autre part à démontrer la capacité de ces méthodes à analyser la réputation d'ArcelorMittal sur divers sujets.

Ainsi, ce mémoire se décline de la façon suivante : la première partie, consacrée à l'état de l'art, propose de poser le sujet et le vocabulaire avant de se pencher sur les travaux existants. Certaines méthodes et algorithmes employés pour qualifier l'opinion selon l'échelle d'étude : celle du document ou celle des aspects (le niveau de granularité étant alors les mots ou les syntagmes) sont abordés. Une seconde partie est consacrée à la présentation de la méthodologie déployée. On y trouve en premier lieu l'évocation de la protection des données puis la constitution du corpus au moyen d'une API *Twitter*. Une troisième partie se concentre sur la présentation et l'analyse du corpus. Enfin la dernière partie est réservée à la présentation des résultats avec le détail des expériences réalisées.

Première partie

Contexte général

CHAPITRE 1

ÉTAT DE L'ART

Sommaire

1.1 Introduction

« The beginning of wisdom is to call things by their proper name. » – Confucius.

L'analyse de sentiments, l'analyse d'opinions, l'analyse d'émotions, la fouille d'opinions,... Autant de termes qui pourraient être considérés comme synonymes étant donnée la frontière parfois floue de leur définition.

Une opinion est une « Manière de penser sur un sujet ou un ensemble de sujets, jugement personnel que l'on porte sur une question, qui n'implique pas que ce jugement soit obligatoirement juste. » - définition tirée du *CNRTL*.
Pour donner ou du moins souligner la structure non apparente que porte un texte (cf. Figure 1.1 [1]), nous donnons cette fois la définition proposée par Liu dans [Liu and Zhang, 2013] : une opinion est un quintuplet $(e_i, a_{ij}, oo_{ijkl}, h_k, t_l)$ où :

- e_i est l'entité (objet, produit, personne, ...) sur laquelle l'opinion est portée,
- a_{ij} est un aspect de e_i, une de ses propriétés,
- oo_{ijkl} est une orientation de l'opinion au sujet de l'aspect a_{ij} de l'objet de e_i. oo_{ijkl} peut être de polarité positive, négative ou neutre ou être exprimée plus finement intégrant alors des niveaux d'intensité (très positif, très négatif,...),
- h_k est le titulaire de l'opinion,
- t_l est le moment (temps) où l'opinion est exprimée par h_k.

1. Illustration tirée du cours suivant https://eric.univ-lyon2.fr/~ricco/cours/slides/WM.A%20-%20Opinion%20mining%20and%20sentiment%20analysis.pdf

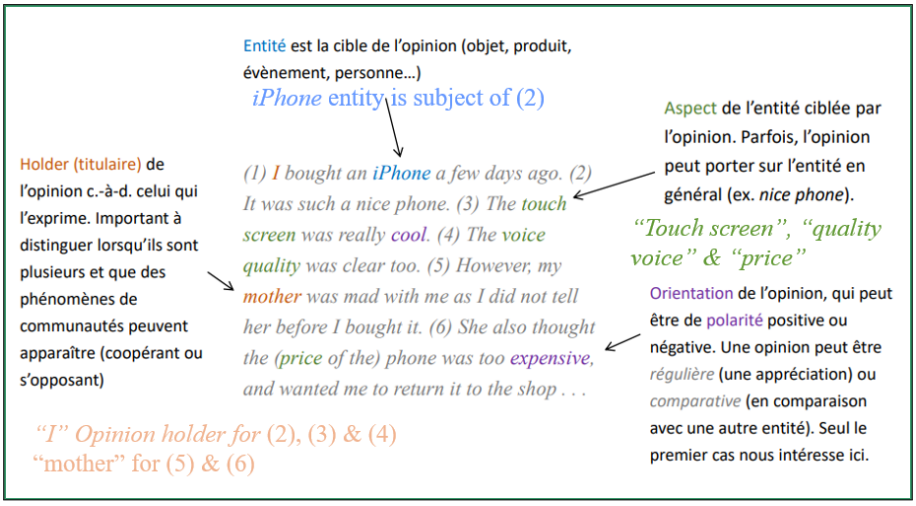


FIGURE 1.1 – Exemple illustrant la définition de l'opinion de Liu

L'objectif de l'analyse d'opinions ou de sentiments consiste étant donnée une collection de documents D à déterminer l'ensemble des quintuplets d'opinions. Cela exige de mettre en place 5 différentes tâches :

— Tâche 1 - Extraction des entités et leur regroupement : il s'agit d'extraire toutes les expressions d'entités de l'ensemble D, et de regrouper toutes les entités synonymes dans un cluster. Chaque cluster d'entité indique une unique entité e_i.
— Tâche 2 - Extraction d'aspects et leur regroupement : il s'agit d'extraire toutes les expressions d'aspects des entités et de les assembler dans des clusters. Chaque cluster d'aspects d'une entité e_i indique un unique aspect a_ij.
— Tâche 3 - Extraction du titulaire de l'opinion et du temps
— Tâche 4 - Classification du sentiment de l'aspect : Il s'agit de déterminer si l'opinion sur un aspect est positive, négative ou neutre (voire plus fine).
— Tâche 5 - Génération des quintuplets d'opinions : A partir des résultats obtenus sur les tâches 1 à 4, construire les différents quintuplets de l'ensemble des documents D.

S'intéresser à l'analyse de sentiments nécessite cependant de considérer des concepts importants telles que la *subjectivité* et l'*émotion*. Pour des raisons de simplification du sujet, ces notions ne sont pas distinguées. Un énoncé objectif expose des faits réels tandis qu'une opinion subjective reflète un sentiment, une émotion, une humeur : elle est donc relative à l'énonciateur...

Dans la littérature, l'analyse de sentiments a été étudiée principalement sur trois niveaux [Patil and Pratibha, 2016] :

— au niveau du document : l'analyse de sentiments classe l'opinion du document entier en un sentiment positif, négatif ou neutre (pour un produit ou un service).
— au niveau de la phrase : l'analyse de sentiments détermine si la phrase exprime une opinion positive, négative ou neutre (pour un produit ou un service). Généralement utilisé pour les documents qui tiennent en une phrase.
— au niveau des entités et des aspects : il s'agit d'identifier et d'extraire les

caractéristiques du produit pour ensuite associer un sentiment à l'aspect.

Dans notre étude nous nous intéressons à l'échelle du document qui est celle du tweet dans une première partie, puis à l'extraction des entités et aspects dans une seconde partie. Différentes approches existantes sont généralement mises en place. La Figure 1.2 liste deux principales approches pour l'analyse des sentiments : une première basée sur le lexique (*lexicon-based*), une seconde sur l'apprentissage machine. A celles-ci, s'ajoute de plus en plus souvent une approche dite *hybride*. Cette approche combine les deux premières.

1.2 Différentes approches de classification

Approches basées sur le lexique (Lexicon-Based Approach)

Ces approches utilisent un lexique de mots qui dénotent un sentiment (*sentiment-word* ou *opinion-word*). Il peut exprimer soit une opinion positive soit une opinion négative *« beau »*, *« bon »*, *« mauvais »*, *« cost someone an arm and a leg »*, etc. La compilation ou constitution de telles listes passe par trois principales approches : une approche manuelle, une approche basée sur le corpus (*« Corpus-Based approach »*) qui implique l'exploitation des statistiques de co-occurence ou de patterns syntaxiques, et une approche basée sur les dictionnaires (*« Dictionary-Based approach »*) qui consiste à exploiter les ressources lexicales et les dictionnaires telles que *WordNet, Merriam-Webster*, etc. [Liu and Zhang, 2013] & [Darwich et al., 2019]. L'approche manuelle est une approche semi-automatique de construction du lexique qui nécessite un check final pour filtrer les erreurs produites.

Approches d'apprentissage automatique

Ces approches se divisent en deux branches : l'apprentissage supervisé et l'apprentissage non supervisé. L'apprentissage supervisé nécessite des données annotées. Ces données permettent de générer un ensemble d'entrainement et un ensemble de test, d'entrainer ensuite un modèle qui permettra alors de prédire des résultats. Les classifieurs les plus couramment utilisés pour l'apprentissage supervisé sont les Arbres de décision (DT - Decision Tree), les SVM, les réseaux de neurones (NN), le Naïve Bayes et le classifieur Maximum Entropy [Ravi, 2015]. La plupart des chercheurs ont conclu que le SVM a une grande précision par rapport aux autres algorithmes (Naïve Bayes, DT & SentiWordNet based approach). Les performances prédictives des différents algorithmes SVM et Naïve Bayes sont comparables, les performances de l'algorithme de l'arbre de décision sont bien inférieures [Patil and Pratibha, 2016].

La classification des documents vise à assigner une ou plusieurs classes à un document. Souvent un préprocessing des données est nécessaire. Les prétraitements les plus courants sont la tokénisation, le retrait des stopwords (*"mots vides"*), la radicalisation, la lemmatisation, les étiquettes morphosyntaxiques (*POS*), les bi-grammes, les *n*-grammes, la gestion des ponctuations, etc. S'en suit alors la construction des *« features »* utilisés par le solveur pour l'apprentissage selon un modèle à base de sac de mots (BOW, TF-IDF,...), ou d'un plongement des mots *« word embeddings »* (Word2Vec, Glove,...) ou de modèles linguistiques de pointes de type

BERT (*Bidirectional Encoder Representations from Transformers*).

Les réseaux de microbloggings dont *Twitter*, limitent le nombre de caractères de leurs tweets. Ces tweets ont une syntaxe particulière faisant notamment apparaître des formules condensées portées par des HashTags *#hashtags*, mais aussi par des arobases *@mention*. Leur syntaxe ne répond pas à une syntaxe *« classique »*. Leur segmentation est un sujet de recherche [Magué et al., 2020] qui se concentre en particulier sur les émojis et les connecteurs.

La classification des sentiments appelée également *détermination de la polarité* vise à déterminer la polarité du document, c'est-à-dire à définir si le document exprime un sentiment positif, négatif ou neutre. Souvent le problème est réduit à une classification binaire, il peut cependant devenir multi-classes en introduisant plus de finesse dans son évaluation à l'instar des évaluations étoilées.

Les techniques d'apprentissage supervisées sont relativement plus performantes que les méthodes non supervisées. Néanmoins, elles exigent un volume de données à étiqueter important (à comparer au volume du corpus) ce qui est très coûteux. Le manque de données annotées pour entrainer les modèles conduit à basculer vers les méthodes non supervisées pour développer des applications. Ces méthodes permettent d'explorer des sujets (*« topics »*) plus finement à l'échelle des aspects (*« multi-aspect fine grained »*) et les sentiments associés. L'apprentissage profond (*Deep Learning*) est utilisé pour définir le niveau de hiérarchie des caractéristiques. La technique a été appliquée dans [Abdullah et al., 2017] aux commentaires d'Amazon pour classifier la polarité des sentiments. Les résultats ne sont pas supérieurs au SVM.

L'approche hybride implique différents types d'approches de classification. L'approche la plus prometteuse dans [Abdullah et al., 2017] combine une approche *lexicon-dictionary based* au SVM dans le cadre de la classification des sentiments de l'inclinaison politique de tweets malais. Les limitations de ces méthodes se trouvent dans le bruit généré [D'Andrea et al., 2015].

Nous comprenons bien que l'idéal serait de mettre en place des techniques performantes sur un corpus volumineux et annoté. Ce qui par ailleurs serait facile à mettre en œuvre. Malheureusement si il est aisé de constituer un corpus volumineux, l'annotation quant à elle reste un véritable goulot d'étranglement. Nous avons donc fait le choix dans cette étude de s'affranchir de l'annotation ce qui bien sûr complexifie le sujet. Mais à ce stade, cela permettra de penser à l'agencement futur d'un pipeline suivant les résultats que cette ébauche apportera. Néanmoins l'annotation futur du corpus n'est pas écartée : il s'agira alors d'annoter de la façon la plus judicieuse en sélectionnant des documents particuliers afin de la rendre la plus efficace possible. Nous continuons cet état de l'art en nous dirigeant vers des librairies et des méthodes qui ont été ciblées comme pouvant nous aider dans ces tâches.

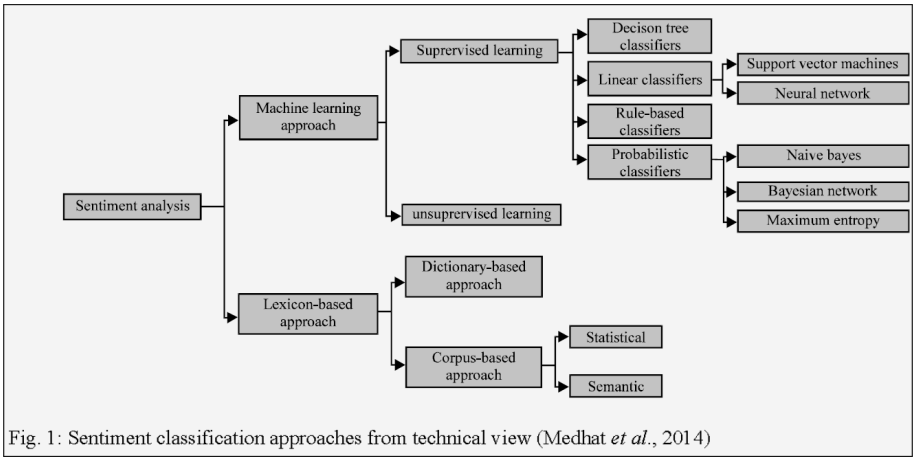


FIGURE 1.2 – Vue d'ensemble des différentes approches pour l'analyse de sentiments tiré de [Abdullah et al., 2017]

1.3 TextBlob

TextBlob [2] est une librairie Python pour le NLP qui utilise NLTK (Natural Language ToolKit). Elle fournit une API simple permettant d'effectuer des tâches telles que le POS tagging, l'extraction de syntagmes nominaux, l'analyse de sentiments, la classification, la traduction, etc. [Khan et al., 2020]

TextBlob est basé sur une méthode de type dictionary-based [Bonta et al., 2019]. L'analyse de TextBlob se fait à l'échelle de la phrase. Le dataset d'entrée est donc segmenté dans un premier temps. Un document est représenté par un sac de mots. Déterminer la polarité consiste à attribuer des scores individuels à chaque mot, le sentiment final est calculé en prenant la moyenne de tous les sentiments. Cet outil permet aussi de fournir la subjectivité. La polarité se situe dans l'intervalle [-1,1], -1 définissant un sentiment très négatif, 1 un sentiment très positif et 0 un sentiment neutre. TexBlob considère et traite les émojis, les ponctuations comme le point d'exclamation par exemple qui intensifie le sentiment. La subjectivité se situe dans l'intervalle [0,1], 0 étant le plus objectif, l'information est factuelle, et 1 le plus subjectif, l'opinion est personnelle. Un paramètre supplémentaire est l'intensité. C'est à partir de l'intensité que la subjectivité est calculée.

Dans [Khan et al., 2020], une analyse de sentiments sur la *Covid-19* est effectuée à partir de données de Twitter. Le focus est fait sur la réaction des gens envers les décisions prises par le gouvernement ou les autorités locales. Un prétraitement est tout d'abord appliqué (segmentation des documents en phrases, tokenization, retrait des stopwords, stemming, quelques regex mises au point pour éliminer les URLs, pour rempacer les mentions *@mention* en le mot mentionné *mention* sans arobase, idem pour les HashTags, puis normalisation en passant en lowercase. Les émoticônes sont remplacés par un mot spécifique qui exprime l'émotion portée par l'émoji, les Re-Tweets c'est à dire les tweets qui sont de nouveau partagés sont conservés mais la mention *RT* qui les identifie supprimée). Une simple utilisation de l'outil TexBlob fournit alors la classification de chaque tweet. Ces données ont finalement constitué

2. https://textblob.readthedocs.io/en/dev/quickstart.html#sentiment-analysis

les input d'un classifieur de type Naive-Bayes.

Dans [Bonta et al., 2019], une analyse comparative de TextBlob, VADER (cf. section 1.4) et NTLK (ses scores de sentiments sont calculés à partir de *SentiWordNet*) est faite sur des données de critiques de films fournis par l'Université de Cornell. L'étude montre une proximité de la distribuition des classifications entre TextBlob et VADER. En terme d'évaluation métrique (cf. précision A.2, rappel A.1 & F-mesure A.3) VADER se place en première position avant TextBlob et bien avant NLTK. Il est constaté dans l'article que NLTK et TextBlob présentent une concentration d'avis incorrects classés comme neutres. La raison soupçonnée est le manque de couverture du langage spécifique aux réseaux sociaux dont les émoticônes, l'argot et la présence d'acronymes.

Dans [László and Attila, 2021] l'objectif est de former un modèle de prédiction des émotions en recherchant les corrélations entre les mots et en les étiquetant selon une décomposition des émotions plus fine que la décomposition habituelle "positif/négatif" pour une prévision plus précise. Une phase d'annotation essentielle a permis d'étiqueter des tweets pertinents de la polarité appropriée. Le corpus est extrait de Twitter avec pour mot-clé *covid* : il s'agit d'analyser l'ambiance régnante. L'analyse se fait en focalisant sur un topic particulier et des corpus de taille réduites (allant de 20 à 500 tweets). Le modèle a été construit et entrainé en se basant sur un Réseau de Neurones Récurrents (RNN) avec utilisation de Keras/TensorFlow (model Sequential classique, avec ajout de plusieurs couches : Embedding initialisé avec des poids aléatoires en premier, Bidirectionel wrapper, couche Dense et Dropout). Parmi les attentes, la reduction du nombre de tweets classés *neutres*. Les résultats du modèle construit sont comparés à ceux de TextBlob. Les résultats montrent effectivement des distributions différentes et largement revues. Le modèle créé arrive à décider d'une polarité non neutre avec 0% là où TextBlob arrive à afficher parfois 45% de tweets neutres.

Dans la section suivante, nous présentons VADER qui connaît le même fonctionnement que TexBlob, et évoquons en particulier la méthode d'Allocation de Dirichlet Latente qui a été utilisée afin de faire émerger des thématiques et des aspects. Ces tâches regroupent partiellement les *Tâches 1 & 2* vues plus haut (cf. 1.1). La méthode étant non supervisée, elle pourrait être d'une grande aide quant à l'extraction d'aspects dans notre cas.

1.4 VADER

VADER (Valence Aware Dictionary and Sentiment Reasoner) est un outil d'analyse de sentiments basé sur un lexique et des règles (*rule-based*). Le lexique VADER est connu pour être performant dans le domaine des médias sociaux, des critiques de films et de produits [Bonta et al., 2019]. Les développeurs de VADER ont utilisé la plateforme web de crowdsourcing *Amazon Mechanical Turk*. Son lexique de sentiments est dit de qualité *« gold-standard »* : il a été validé par des humains. Comme pour TextBlob, il ne nécessite pas de données annotées. Lors de l'analyse d'un texte, VADER cherche si un mot appartient au vocabulaire. VADER renvoie le score composé, une métrique qui calcule la somme de toutes les évaluations (positives, négatives et neutres) du lexique normalisée. Un seuil de 0.05 permet de définir les inter-

valles utiles à la classification d'un document. Ainsi un score composé appartenant à l'intervalle [-1,-0.05] classe le document comme négatif, un score appartenant à]-0.05,0.05[classe le document comme neutre et un score appartenant à l'intervalle [0.05,1] classe le document comme positif. -1 et + 1 expriment les sentiments les plus extrêmes : très négatif et très positif respectivement. Une métrique pour chaque classe est fournit également.

1.5 LDA

Dans l'article [Khairiyah Mohamed and Carol Anne, 2021] une analyse de sentiments et une modélisation des thématiques de discussions sur les tweets concernant Covod-19 à Singapour sont faites. L'analyse des sentiments a été effectuée en utilisant VADER, celle des émotions en utilisant un RNN pré-entrainé. La modélisation des topics a été traité avec une méthode Gibbs Sampling Dirichlet Multinomial Mixture (GSDMM) mais aussi avec une Allocation de Dirichlet Latente pour la définition d'une borne supérieure du nombre de topics pour le modèle GSDMM. Les résultats de ces deux méthodes ont alors été imbriqués de sorte à analyser les sentiments associés à chaque topic. Une première conclusion est l'importance d'analyser le sentiment par topic/thème pour mieux comprendre un tweet.

La Figure 1.3 est une illustration tirée de l'article [Amara et al., 2020] qui schématise et présente le modèle génératif probabiliste LDA.

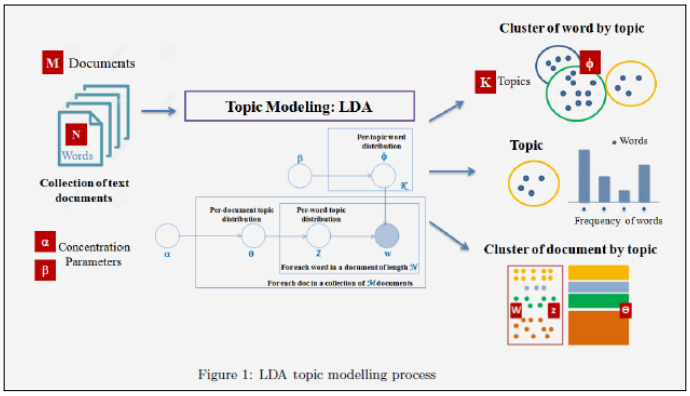


FIGURE 1.3 – Mécanisme de LDA topic modeling

1.6 CRF & Bi-LSTM pour ATE [explicites]

On s'intéresse ici à l'analyse au niveau de l'aspect (*Tâche 2* cf. 1.1).

N.B. : La technique n'a pas été développée dans ce rapport. Il s'agit d'une des dernières avancées récentes mettant en œuvre des réseaux de neurones bi-LSTM (bi-directionnal Long Short-Term Memory) et un CRF (Conditional Random Fields) pour l'extraction d'aspects. Nous prévoyons d'aller dans ce sens dans le futur cependant la méthode nécessitant un étiquetage, la première étape reste à construire tous les éléments qui serviront à nourrir ces modèles.

On appelle une analyse de sentiment à l'échelle des aspects *Aspect Based Sentiment Analysis (ABSA)*. C'est un des domaines les plus dynamiques en NLP actuellement.

Cette analyse se décompose en 4 sous tâches : l'extraction des termes d'aspect, la polarité des termes d'aspect, la détection des catégories d'aspect et la polarité des catégories d'aspect (cf. Figure 1.4). L'article suivant [Gandhi and Attar, 2020] s'intéresse en particulier à la première sous tâche d'ABSA à savoir l'extraction de termes d'aspect ou *Aspect Term Extraction (ATE)*. L'article traite la manière dont l'ATE peut être réalisée sur des textes de langue dont la morphologie est riche tel que le Hindi. Les modèles proposés sont d'une part un CRF, d'autre part un Bi-LTSM basés sur une approche nouvelle prenant en considération des features complémentaires. Un *ClusterId* pour le CRF, le *POS tagging* pour le réseau Bi-LSTM. Les résultats pour le CRF sont améliorés passant d'une F-mesure de 41.07% à 42.71% puis atteignant 44.54% via une 5-fold cross-validation dépassant les résultats que l'on trouve dans la littérature. Le modèle Bi-LSTM avec un POS-vecteur obtient une F-mesure de 44.49%.

N.B. : La Figure 1.4 évoque des 'catégories' alors qu'il s'agit essentiellement d'hyperonymes, d'une ontologie. Les 4 aspects sont plutôt d'ordre explicite d'une part, et d'ordre implicite d'autre part. L'article suivant [Abdi et al., 2020] traite aussi bien des extractions explicites que des extractions implicites en proposant de nombreuses techniques (supervisées, non supervisées, semi-supervisées avec une distribution des modèles CRF, LSTM, etc.) tout en résumant les résultats de 2008 à 2019.

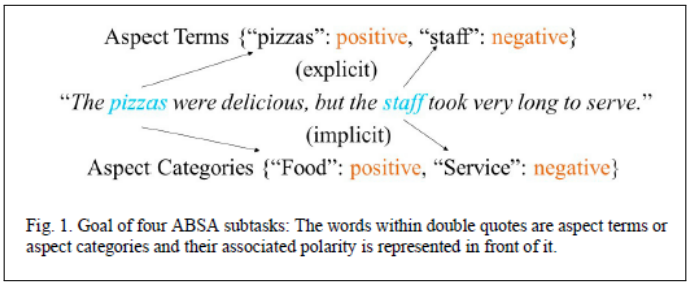


FIGURE 1.4 – Représentation des sous-tâches de l'ABSA tirée de [Gandhi and Attar, 2020]

La méthodologie proposée pour la tâche d'extraction, après un pré-traitement, consiste à tout d'abord annoter les aspects d'un schéma *BIO*. Chaque mot w_{ij} d'une phrase r_i est assigné d'une étiquette $l_k \in L$ où L est l'ensemble représentant les labels BIO soit $\{B : Begin, I : Inside, O : Outside\}$.

— Pour le CRF, les features du modèle sont : les mots et leur contexte (avec une fenêtre de taille fixe : unigramme, bigramme et trigramme), les POS des mots et leur contexte avec la même fenêtre, pour le cas du Hindi des informations concernant les préfixes et suffixes (ce dont nous n'aurons pas besoin dans notre cas), une information signalant le mot "de tête" (le premier mot si l'ATE est constitué de plusieurs mots) et le *cluster ID* obtenu suite à une clusterisation utilisant Kmean réduite de 70% du word embedding issu de Word2Vec avec un skip-gram (cf. Fig. 1.5).
— Pour le Bi-LSTM, les features du modèle sont : une couche d'embedding (créée à partir du vocabulaire et d'un modèle pré-entrainé Glove) concaténée à un POS-vecteur (qui est un one hot vecteur de POS) (cf. Fig. 1.6).

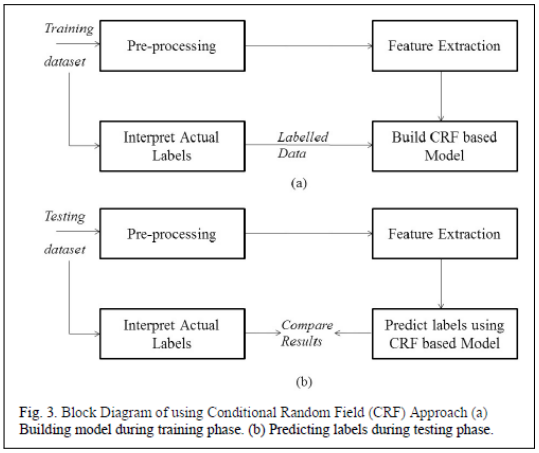


Fig. 3. Block Diagram of using Conditional Random Field (CRF) Approach (a) Building model during training phase. (b) Predicting labels during testing phase.

FIGURE 1.5 – ATE avec CRF [Gandhi and Attar, 2020]

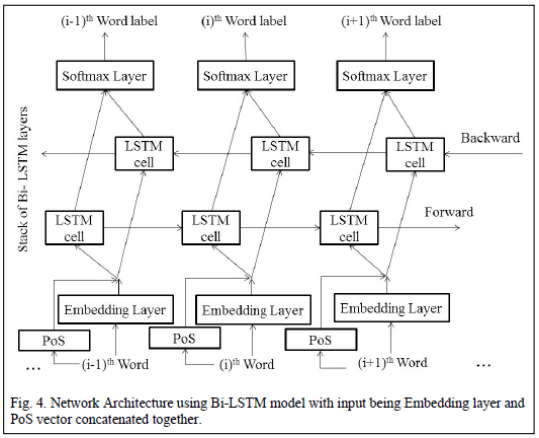


Fig. 4. Network Architecture using Bi-LSTM model with input being Embedding layer and PoS vector concatenated together.

FIGURE 1.6 – ATE avec Bi-LSTM & Embeddings & POS Vector [Gandhi and Attar, 2020]

1.7 BERT

BERT (*Bidirectional Encoder Representations from Transformers*) est un modèle de Deep Learning pré-entrainé. Il a été lancé par *Google AI* en octobre 2018 [Vaswani et al., 2017]. BERT en comparaison de Word2Vec génère un plongement de mots qui prend en compte le contexte. C'est le mécanisme d'attention (cf. Fig. 1.7) mis en œuvre par les encodeurs qui assure ce comportement. Chaque token est remplacé par une projection de son vecteur d'embeddings de plus petite dimension (*queries ,keys, values*). La première étape consiste à calculer le produit scalaire entre ces pairs d'embeddings (*queries, keys*). Lorsque les vecteurs sont fortement corrélés ou similaires le produit scalaire est élevé, les tokens ont alors un lien fort, dans le cas contraire les mots n'ont pas de liens. Les valeurs sont ensuite normalisées et fournies en entrée à une fonction d'activation de type *softmax* qui amplifie les valeurs. La dernière étape consiste à former le nouveau vecteur d'embeddings qui prend en compte le contexte via une combinaison linéaire des embeddings d'entrée projetés (*values*) et les poids fournis par les résultats du softmax. Le même processus répété sur différents couples (*queries, keys*) est appelé *multi-head attention* (cf. Fig. 1.8). Ces techniques sont embarquées dans BERT (cf. Fig. 1.9 [3]).

Depuis la sortie de BERT, plusieurs modèles pré-entrainés dérivés ont vu le jour. La communauté de chercheurs est très dynamique, de nombreux modèles pré-entrainés pour des tâches NLP données sont développés et partagés sur des plateformes open-source en particulier *Hugging Face*. Nous avons testé plusieurs versions dans le cadre de ce travail, elles sont présentées au fur et à mesure de leur utilisation dans le *Chapitre 4*.

Pour une grande partie des tâches en NLP, BERT décroche les meilleurs résultats (cf. [Lagrari and Elkettani, 2021] & [Mrityunjay et al., 2021] par exemple). Ces dernières années divers systèmes ont été développés en utilisant BERT pour l'analyse de sentiments. L'article suivant [Colón-Ruiz and Segura-Bedmar, 2020] compare

3. Illustrations de Romain Futrzynski https://peltarion.com/blog/data-science/self-attention-video

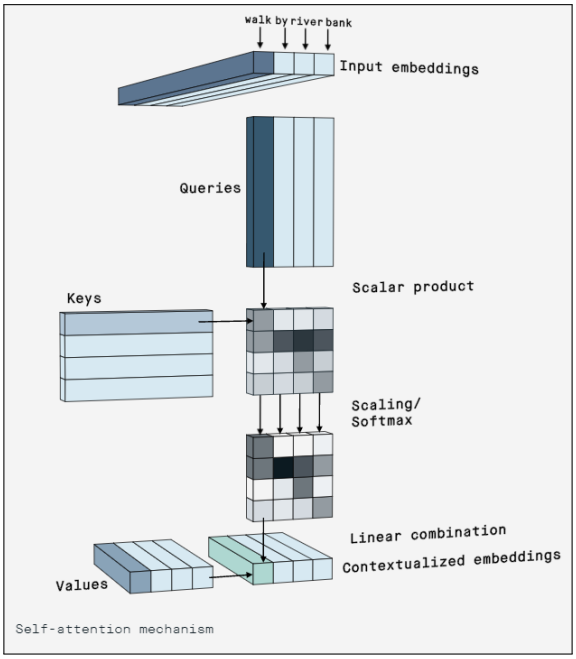


FIGURE 1.7 – Mécanisme de self-attention

des architectures de Deep Learning pour l'analyse de sentiments sur des textes d'évaluations de médicaments. Parmi les architectures proposées, on trouve : un simple CNN, un Bi-LSTM, une combinaison d'architectures utilisant un CNN et un Bi-LSTM (CNN suivi d'un Bi-LSTM, CNN concaténé à un Bi-LSTM, ou un Bi-LSTM suivi d'un CNN) et un modèle basé sur BERT et un LSTM. BERT fournit quelques soient les expériences en terme de nombre de classes, les meilleurs résultats.

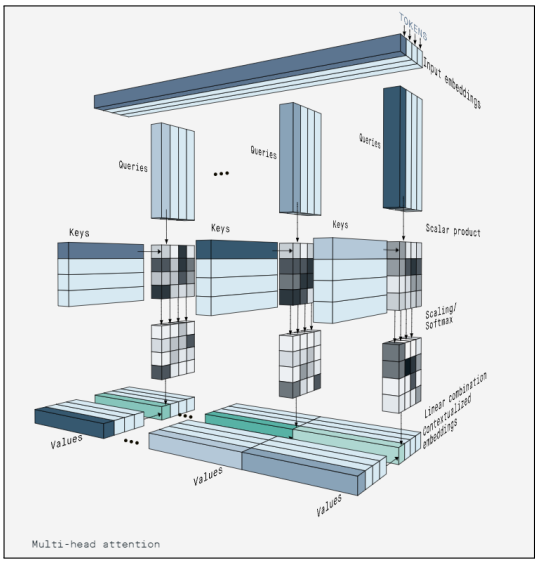


FIGURE 1.8 – Mutlihead Attention

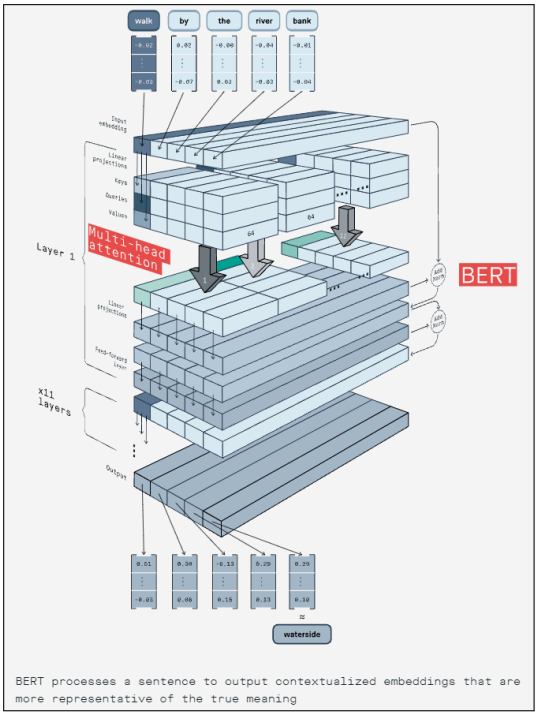


FIGURE 1.9 – Principes de BERT

1.8 Conclusion

Pour conclure, dans [Lagrari and Elkettani, 2021] les auteurs comparent l'application des approches dites traditionnelles (celles reposant sur des approches lexicon-based et Machine Learning) aux approches de Deep Learning pour l'analyse des sentiments. Pour les approches traditionnelles, les techniques sont de plus en plus hybrides et combinent machine learning et orientation sémantique de sorte à suivre le mouvement dynamique du langage (structure plus complexe, textes courts, etc.). Les approches de Deep Learning quant à elles s'attachent à répondre à ces traits. L'article présente un tableau qui résume et compare les performances de divers techniques de Deep Learning pour des tâches d'analyse de sentiments. Le détail de la tâche n'est pas indiqué cependant les articles sont référencés. Les meilleurs résultats reviennent à BERT et ses dérivées.

A ce stade, nous avons plusieurs pistes à étudier, afin d'évaluer ce qu'il serait possible d'extraire de ces méthodes dans le cadre de notre étude.

CHAPITRE 2

MÉTHODES

Sommaire

2.1 Introduction

L'objective de l'étude est de mesurer la perception d'ArcelotMittal à travers les médias. Les tendances peuvent bien évidemment évoluer selon le sujet, la zone géographique, l'échelle de temps, les groupes (étudiants, universités, salariés de la société, industries, concurrents, syndicats, politiques, etc.), les sites de presses en ligne, etc.

Plusieurs réseaux sociaux, sites webs et magazines ont été initialement ciblés pour l'extraction d'informations. Cependant, il a été proposé pour des raisons de simplification et d'introduction à l'étude, de restreindre le périmètre. Aussi le Règlement Général sur la Protection des Données (RGPD) a dirigé notre travail. La première partie de nos travaux à consister à construire un corpus, la seconde à chercher, à étudier et à tester des méthodes qui répondent à l'analyse de sentiments à l'échelle du document d'une part, puis à l'échelle des entités et aspects d'autre part.

2.2 Constitution du corpus

Notre choix s'est porté sur le réseau *Twitter*. Nous avons utilisé son *API standard V1.1*. Il est à noter qu'elle limite aux tweets des 7 derniers jours. Il n'était pas possible selon les règles d'utilisation de passer à une version *Academic Research* qui donne accès à une quantité et un éventail de données plus importants. Néanmoins, Twitter fournit des éditions *Standard*, *Premium* et *Enterprise* pour ses APIs. Les APIs standard sont gratuites et généralement lentes par rapport au Premium ou au Enterprise.

Pour pouvoir accéder aux tweets, il faut dans un premier temps créer un compte développeur *« Twitter Developer Account »* et enregistrer son application pour accéder

aux APIs Twitter. Dans un second temps, il faut répondre à des questions qui demandent de fournir des informations expliquant le besoin d'accès aux données. Une fois toutes les étapes validées, des clés personnelles *API access key* et *API secret key* sont fournies. Il suffit alors de donner un nom à l'application pour accéder au *Access token* et *Access Token Secret key* : ces clés sont nécessaires pour l'authentification et donc l'accès aux données. L'utilisation de l'API peut alors se faire en utilisant *searchtweets* ou *Tweepy*. Le code *twitterCorpusStand7Days.ipynb* permet l'extraction des tweets.

N.B. : Concernant Twitter API standard V1.1, il n'est pas possible d'utiliser les données issues de Twitter à des fins commerciales et avec des tiers. Si les données Twitter sont utilisées aux fins d'amélioration des produits ou d'image de marque alors il s'agit de fins commerciales. Ces restrictions figurent à l'article II.B + II.C du « Developer Agreement and policy » [1].

Remarques :

— Dans le cas de notre étude, nous avons pu constater la limitation engendrée par la version Standard V1.1. L'impact sur le volume du corpus est majeur. En effet, la Figure 2.1 présente le nombre de tweets extraits sur la période du 14/08/2021 au 20/08/2021 pour des requêtes différentes. Ainsi une requête avec le mot clé *pfizer* retourne *254828* tweets tandis qu'une requête avec le mot clé *arcelormittal* renvoie *329* tweets.
— La limitation dans le temps peut accentuer les faits d'actualité : ce qui peut expliquer le nombre de tweets pour *pfizer*.

keyWord	Tweet Nb
arcelormittal	329
vaccins	825
vaccin	1693
pfizer	254828

FIGURE 2.1 – Comparaison du volume de données extraits sur une même période selon le mot clé de la requête

D'autres versions d'APIs Twitter, *SandBox 30 days* et *SandBox FullArchive*, ont été testées. Le nombre de tweets a donc augmenté mais les limitations restent fortes et induisent la constitution d'un petit corpus.

Une fois le corpus généré, nous nous sommes d'abord familiarisés à ce dernier en nous inspirant de l'*Analyse des Données Textuelles (ADT)* [Valette, 2016]. Nous nous sommes intéressés en particulier à la distribution des tokens et à leur occurence.

En deuxième lieu, nous sommes entrés dans le vif du sujet à savoir l'analyse de sentiments.

1. https://developer.twitter.com/en/developer-terms/agreement-and-policy

2.3 Méthodes d'analyse de sentiments à l'échelle du document

Une première étude a été faite à l'échelle du document.

Inspirés des articles lus, l'idée était de mettre en place un *bootstrapping*. Celui-ci permettait alors de s'affranchir de la rédaction d'un guide d'annotations et de l'annotation manuelle, et de construire les données d'entrée nécessaires à un modèle supervisé. Nous avons utilisé les méthodes de type lexicon-based les plus rencontrées : *TextBlob* et *VADER*. Ce choix a été complété par la suite par la sélection de deux méthodes pré-entrainées de Deep Learning de type *BERT*.

Aussi pour s'assurer du bon étiquetage des documents par ces méthodes, nous avons en premier lieu pensé à ne sélectionner pour un apprentissage de modèle que les documents dont l'étiquette allait être commune entre les méthodes. Dès lors les méthodes proposées auraient fourni l'étiquetage nécessaire à une méthode supervisée afin de créer et d'entrainer un modèle qui aurait ensuite permis la prédiction.

Une étude comparative de TextBlob et VADER a finalement remis en cause les hypothèses posées. L'intersection des résultats fournis par ces méthodes n'est pas forcément un bon candidat pour la construction d'un modèle de prédiction.

Un autre axe d'investigation a été l'utilisation de modèles pré-entrainés adaptés à des tâches spécifiques et basés sur BERT [Lagrari and Elkettani, 2021]. Les modèles sélectionnés datent de 2020. Ils ont été pris en main et testé. Les résultats sont intéressants cependant il faudrait approfondir cet axe afin de statuer.

2.4 Méthodes d'analyse de sentiments à l'échelle des topics/aspects

Une seconde étude a été faite à l'échelle des entités et des aspects.

De la même façon que précédemment, nous nous sommes abstenus de l'annotation. Cette fois-ci l'annotation aurait permis la construction directe d'un modèle d'apprentissage de reconnaissance d'entités nommées (REN) en utilisant spaCy V.3 par exemple.

Afin d'extraire les entités/thématiques et/ou aspects dont il est question dans les tweets, nous avons dans un premier temps exploré la piste du modèle génératif Latent Dirichlet Allocation (LDA). Les clusters créés ainsi que leurs mots-clés forment des résultats "bruts" déjà intéressants. Fort de ce constat, une approche similaire exploitant BERT a été explorée à son tour : les résultats semblent prometteurs. A ce stade, un socle se dessine constituant un point de départ pour la construction d'une liste d'aspects et de thèmes.

En supposant la liste des aspects crées, un sujet restant est : définir la polarité qui leur est associée. Nous avons donc entamé un autre champ d'investigation

basé sur l'analyse syntaxique des tweets cherchant à exploiter les arbres de dépendances de sorte à lier ses aspects à une orientation d'opinion.

2.5 Conclusion

En conclusion, nous avons constitué un corpus de petite dimension étant données les restrictions aux quelles nous avons du faire face. Divers champs d'investigations ont été proposés pour répondre d'une part à l'analyse de sentiments à l'échelle du document (approches lexicon-based et Deep Learning de type BERT), d'autres part à l'analyse de sentiments à l'échelle des aspects (pour les aspects : modèle probabiliste génératif LDA VS Deep Learning de type BERT, pour la polarité d'opinion : analyse syntaxique).

Deuxième partie

Expérimentations

CHAPITRE 3

CORPUS

Sommaire

3.1 Introduction

Ce chapitre a pour vocation de nous familiariser d'une part à la structure du corpus et d'autre part à son contenu d'une façon générale. Une cartographie de la distribution des tokens qui le forment est donnée en entrée. Le chapitre se concentre sur deux pôles : les hashtags, qui sont de véritables mots clés dont le contenu est condensé et les tweets. C'est à travers des graphes de distributions des fréquences que certaines thématiques sont dégagées.

3.2 Analyse du corpus

L'API Twitter fournit les tweets en format JSON. Les tweets peuvent compter plus de 150 attributs. Nous nous sommes intéressés à un cercle très réduit de ces données pour répondre à la protection des données personnelles quant à l'identification d'une personne. Nous manipulons donc les données suivantes :

— *created_at* : la date de création du tweet,
— *id_str* : l'identifiant du tweet,
— *text* : le tweet. Cette variable peut contenir jusqu'à 140 caractères. *N.B. : Les tweets sont limités à 280 caractères.*
— *truncated* : un booléen qui indique si le tweet a été tronqué. Vrai si le tweet contient plus de 140 caratères, Faux sinon.
— *favorite_count* : le nombre de *Likes*,
— *entities* : le champ qui contient notamment la liste des *#HashTags* mais aussi la variable *extended_tweet* élément contenant le tweet non tronqué *full_text*.

Notre corpus compte 878 tweets au total. En enlevant les retweets (tweets republiés) seul 480 tweets sont restants. La Figure 3.1 présente un extrait du corpus.

	full_tweet
0	ArcelorMittal Has World's First 'Green' Steel Voucher Program https://t.co/slXiELWjJf https://t.co/On61CSi5Wz
1	@AssindMantova 'A 50/50 joint venture between ArcelorMittal and Nippon Steel Corp. (NSC), the facility is located in Calvert, Alabama, USA.' https://t.co/ibhrj11wlw
2	ArcelorMittal Investing Nearly $400 Million in Eliminating Carbon Emissions https://t.co/vP0fTFSkXW via @thomasnet
3	ArcelorMittal Has World's First 'Green' Steel Voucher\nProgram \nhttps://t.co/VAJdW475lw
4	$MT ArcelorMittal call volume above normal and directionally bullish https://t.co/noHTbKx5Em
5	$MT ArcelorMittal call volume above normal and directionally bullish\nBullish option flow detected in ArcelorMittal with 7,079 calls trading, 1.4x expected, and implied vol increasing almost 3 points to 52.03%. Jan... https://t.co/7DUkZAh2or
6	"It is awesome! It was wonderful experience and I'd recommend it to any thrill junkie like myself." – Rubina, from Nottinghamshire \n\nTake on the ArcelorMittal Orbit Abseil, like Rubina, for an experience you'll never forget: https://t.co/Adf9xMEiVB https://t.co/Y3bDHRbw8R
7	The steel industry continues to negotiate with governments about how best to clean up their operations, but the clock is ticking #aLotToDo #HugeChallenge #ThinkGreen\nhttps://t.co/FTVYtSjsBm
8	Check out how ArcelorMittal is improving efficiency and streamlining operations with machine learning technology enabled by the Intel Distribution of OpenVINO toolkit! #IAmIntel #IOT https://t.co/OEekEfXJqu
9	ArcelorMittal, the world's largest steel producer thinks it's possible to make the alloy without also producing millions of tons of carbon emissions — and it's spending $390M in several pilot programs in hopes of proving it. #steel...https://t.co/XYqrQzId4F https://t.co/pAPII7j8uY

FIGURE 3.1 – Quelques tweets extraits du corpus.

Le code correspondant au chargement du corpus et aux analyses des sections suivantes est *preprocessing.py*.

3.2.1 Distribution des tokens

Afin de nous familiariser avec notre corpus, mais aussi afin de nous assurer dans le futur que nous ne perdons aucun élément du corpus lors d'une phase de traitement allant de l'extraction à l'enregistrement en BD tout en passant par des pré-processings, nous avons développé plusieurs fonctionnalités dans le but d'énumérer chaque type de tokens. Par *token* nous entendons toute « unité minimale » la ponctuation en faisant partie. La figure 3.2 fournit la fréquence de distribution de chaque type de token.

Ainsi, nous avons dénombré chaque token (retour à la ligne (\n) inclus, séparation des ponctuations accolées) en conservant la casse : on compte *1*3792 tokens. La considération d'une unique occurence donne *3*933 tokens casse prise en compte.

Pour des raisons de protections des données nous ne présentons pas les *u*ser mention : ils sont au nombre de *5*19, sans répétition ils sont *3*21. La figure 3.3 présente la distribution des tokens selon leur types : @Token pour les users mentions, #Token pour les hashtags, , %Token pour les poucentages, 123Token pour les nombres, alphaToken pour les mots ne comportant que des caractères alphabétiques, $Token pour les devises, etc. Sous le nom *else* on trouve des émoticônes, des éléments chimiques (CO2, H2, ..), des tokens moins structurés (More@, 1-year, etc.). On peut retrouver une extraction des emojis en annexe A.8 : nous avons utilisé advertools pour extraire les emojis et leurs associer un text. *N.B. : Les hashtags n'ont pas été segmentés.*

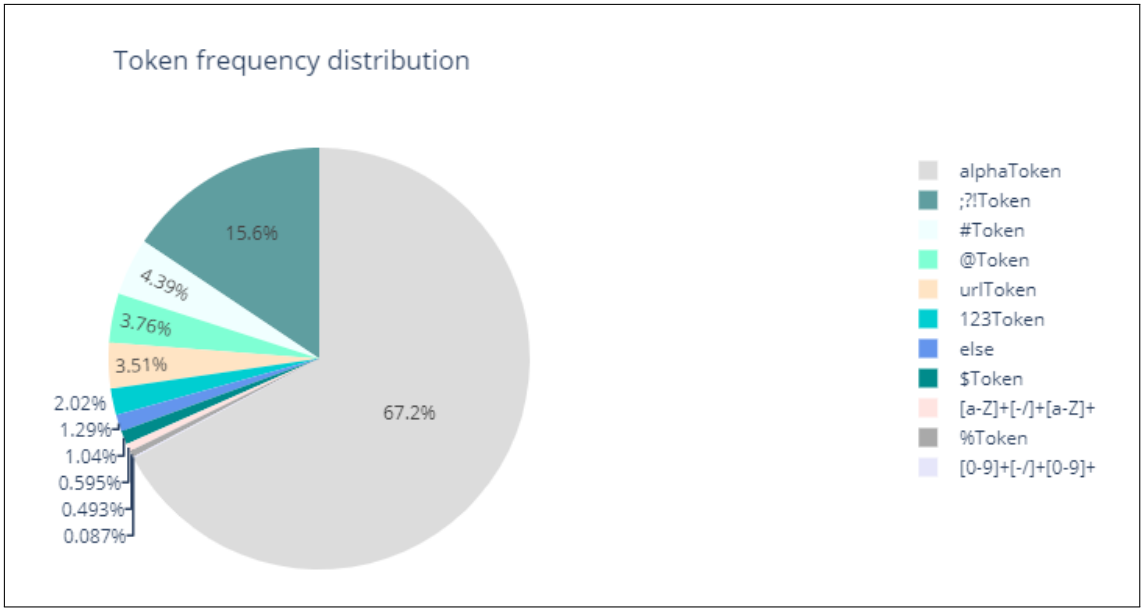


FIGURE 3.2 – Distribution des tokens du corpus.

	TokenType	OccurenceNb
0	@Token	519
1	#Token	606
2	$Token	143
3	%Token	68
4	;?!Token	2155
5	urlToken	484
6	123Token	279
7	alphaToken	9266
8	[0-9]+[-/]+[0-9]+	12
9	[a-Z]+[-/]+[a-Z]+	82
10	else	178

FIGURE 3.3 – Distribution des tokens du corpus.

	[a-zA-Z]+[-/]+[a-zA-Z]-Token	NbOccurences
0	low-carbon	13
1	gigawatt-scale	4
2	long-term	3
3	anti-dumping	3
4	non-alloy	3
5	over-supply	2
6	am/ns	2
7	dri/hbi	2
8	ex-works	2
9	zero-carbon	2
10	fast-track	2
11	ammonium/cyanide	1
12	eye-catching	1
13	net-zero	1
14	north-western	1
15	anti-slip	1
16	h-dr	1
17	worldwide-supplied	1

FIGURE 3.4 – Distribution des tokens composés du corpus.

3.2.2 Nombre de tweets likés

Nous nous sommes tout d'abord intéressés aux "likes". En effet certaines études utilisent les évaluations clients pour définir une target : sur le même principe, nous cherchions donc à voir si la variable était exploitable. La Figure 3.5 présente le nombre de tweets likés. Ainsi on constate que 274 tweets ne sont pas likés, 79 tweets ont été likés une unique fois, 1 tweet a été liké 103 fois, etc. Plus de la moitié des tweets ne sont pas likés : au vu du peu de réactivité des internautes sur le contenu des tweets, cette variable ne sera pas exploitable.

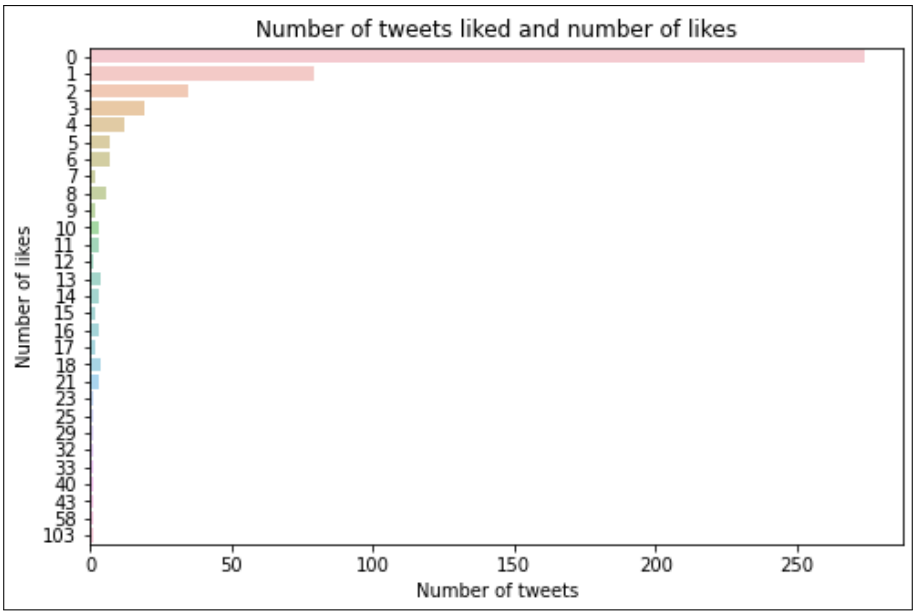


FIGURE 3.5 – Nombre de tweets en fonction du nombre de likes.

3.2.3 Distributions des #Hashtags

Les hashtags sont des mots-clés stratégiques qu'il faut savoir utiliser efficacement : ils permettent d'élargir l'audience en atteignant les internautes qui ne sont ni « amis » ni « followers », ils augmentent le taux d'interactions, etc. Des études de marketing [1] révèlent les bonnes pratiques utiles et les meilleures stratégies pour augmenter l'engagement des internautes. Nous nous intéressons à cette variable présente à deux niveaux dans les données : dans la structure *Entities* et dans le contenu des tweets.

On compte au total sans aucun pré-traitement 192 hashtags (multiples occurences incluses) dans la structure *Entities*. La dimension du vocable associée est de 122. Dans la mesure où un même mot-clé peut être écrit en majuscule ou en minuscule, on s'intéresse à son nombre sans considération de la sensibilité à la casse : il est de 115. La Figure 3.6 présente les 20 plus fréquents hashtags (en lowercase). *ArcelorMittal* et *steel* sont sans surprise les hashtags les plus fréquents. Ils sont suivis d'*hydrogen*. La majorité des hashtags plafonnent à 2 ou 3 occurences : la dimension du corpus y est pour beaucoup.

1. *« The best Hashtag Strategies for Social Media »* http://pages.trackmaven.com/rs/251-LXF-778/images/hashtag-strategies-for-social-media.pdf

La liste des hashtags de la structure *Entities* n'est pas forcément toujours complète : nous avons donc extrait les hashtags des tweets. Le corpus compte 606 hashtags. La dimension de l'ensemble des vocables est de 326. Les Figures 3.7 et 3.8 présentent de la même façon les 20 plus fréquents hashtags avec et sans normalisation. Idem *ArcelorMittal*, *steel* et *hydrogen* se démarquent. La distribution des mots-clés évoluent. Une thématique autour de l'énergie « propre » est commune *hydrogen, carbonneutral, decarbonization, energy, environment, greensteel, etc.*, une thématique autour des technologies et du numérique émergent *imintel, iot*. L'évolution du nombre d'occurences plafonne assez rapidement à 8 occurences cf. les Figures 3.10 et 3.9 qui donnent le nombre d'occurences des 10 premiers hashtags.

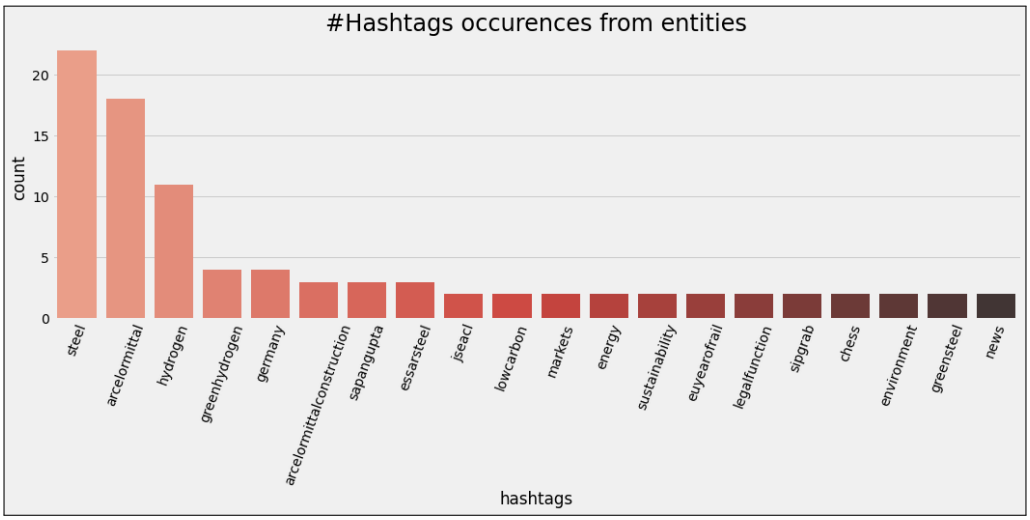


FIGURE 3.6 – Les 20 plus fréquents #hashtags tirés de la structure *Entities*.

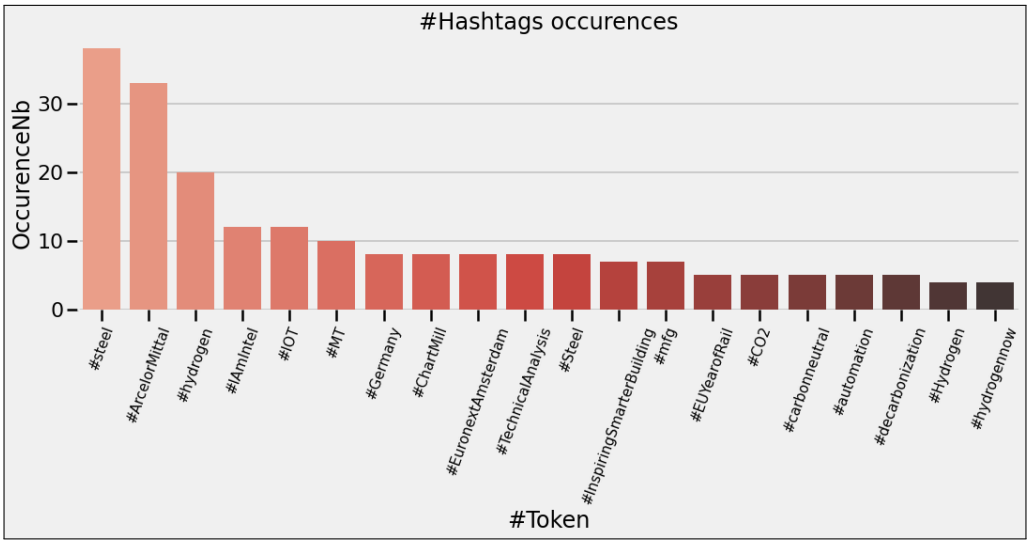


FIGURE 3.7 – Les 20 plus fréquents #hashtags présents dans les tweets.

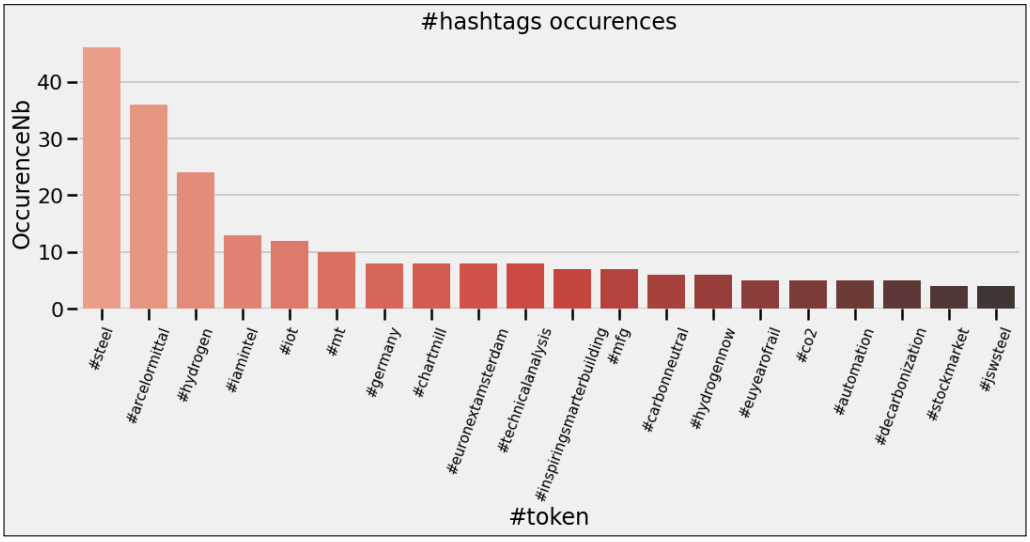


FIGURE 3.8 – Les 20 plus fréquents #hashtags présents dans les tweets en minuscule.

	#Token	OccurenceNb
0	#steel	38
1	#ArcelorMittal	33
2	#hydrogen	20
3	#IAmIntel	12
4	#IOT	12
5	#MT	10
6	#Germany	8
7	#ChartMill	8
8	#EuronextAmsterdam	8
9	#TechnicalAnalysis	8

FIGURE 3.9 – Les 10 plus fréquents #Hashtags dans les tweets.

	#token	OccurenceNb
0	#steel	46
1	#arcelormittal	36
2	#hydrogen	24
3	#iamintel	13
4	#iot	12
5	#mt	10
6	#germany	8
7	#chartmill	8
8	#euronextamsterdam	8
9	#technicalanalysis	8

FIGURE 3.10 – Les 10 plus fréquents #Hashtags dans les tweets en lowercase.

3.2.4 Nuages de mots des #Hashtags

Une autre représentation graphique de la fréquence des hashtags peut être effectuée par l'intermédiaire de nuages de mots. Celle-ci permet d'avoir une vue d'ensemble sur les hashtags les plus pesants dans le corpus. Des sujet majeurs, des filiales ainsi que des mots-clés qui étaient ciblés pour l'étude apparaissent : *XCarb®* , *sustainability, CO2, Environment, etc* cf. Figures 3.11 à 3.14 .

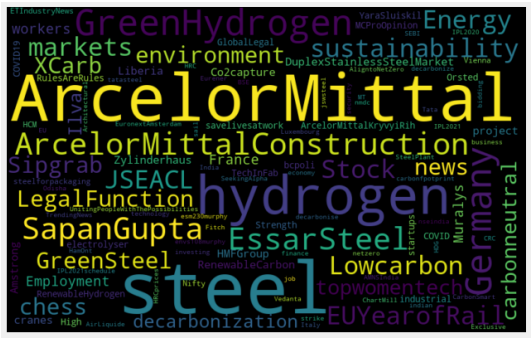


FIGURE 3.11 – Nuage de mots des #hashtags tirés de la structure *Entities*.

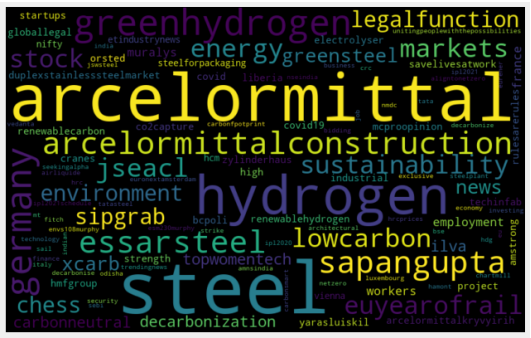


FIGURE 3.12 – Nuage de mots des #hashtags en lowercase tirés de la structure *Entities*.

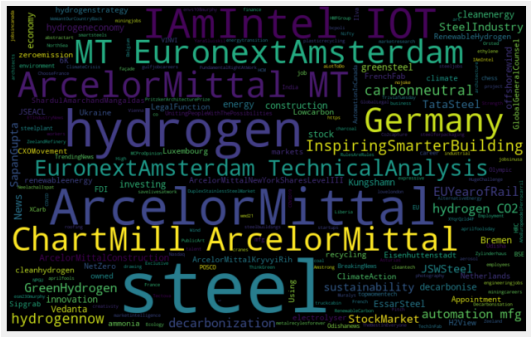


FIGURE 3.13 – Nuage de mots des #hashtags tirés des tweets.

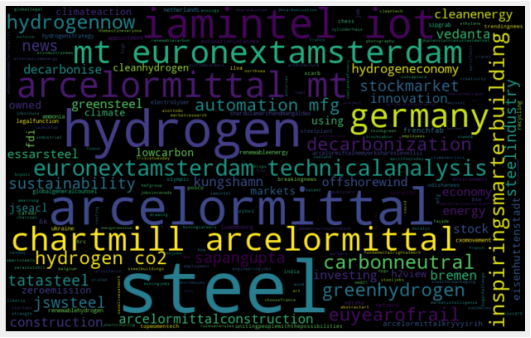


FIGURE 3.14 – Nuage de mots des #hashtags en lowercase tirés des tweets.

3.2.5 Mots les plus fréquents dans le corpus

Dans cette section, nous nous sommes intéressés aux mots les plus représentatifs du corpus (hors hashtags). Pour cela, nous avons élagué le corpus de différents tokens tels que les hashtags, les mentions d'utilisateurs, les digits, les URLs, les ponctuations, les stopwords, etc. Le graphe suivant cf. Figure 3.15 présente les 20 plus fréquents tokens.
Bien entendu *arcelormittal, arcelor, mittal, steel* sont prédominants, des groupes de mots-clés cohérents apparaissent également : un premier tourné vers la production et les usines *(steelmaking, industry, plants, amp (ArcelorMittal Poland), production)*, un second traitant de l'energie et de l'environnement *(green, hydrogen, cleaner, emissions)*.

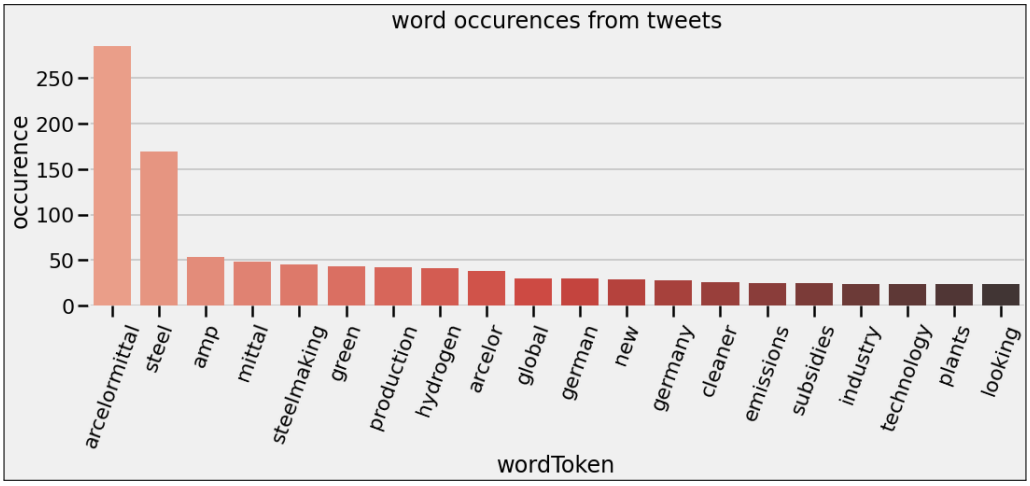


FIGURE 3.15 – Les 20 plus fréquents tokens du corpus.

3.2.6 Nuages de mots du corpus

Une représentation en nuages de mots a également été effectuée sur le corpus.

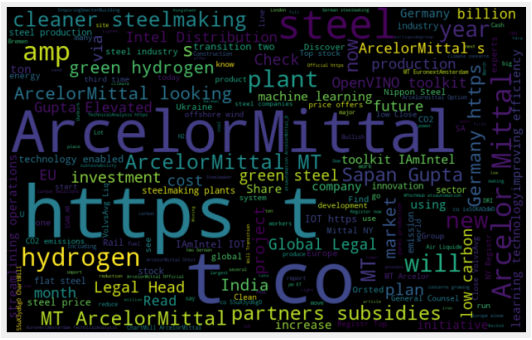


FIGURE 3.16 – Nuage de mots tirés des tweets sans preprocessing.

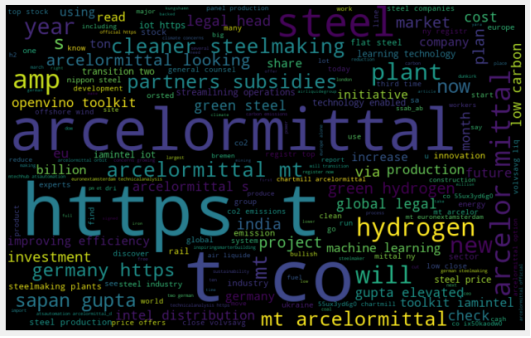


FIGURE 3.17 – Nuage de mots en lowercase tirés des tweets sans preprocessing.

FIGURE 3.18 – Nuage de mots tirés des tweets avec un soft preprocessing.

FIGURE 3.19 – Nuage de mots en lowercase tirés des tweets avec un soft preprocessing.

Les différentes figures cf. Figures 3.16 à 3.19 illustrent les nuages de mots en

fonction de quelques pré-traitements : sensibilité à la casse et élimination des stopwords. Des thématiques émergent à travers des mots-clés/aspects *(green steel, green hydrogen, plants green, cleaner, emissions, etc.)*, *(openvino, machine learning, , learning technology, intel, etc.)*, etc.

3.3 Conclusion

L'analyse du corpus, bien que peu rencontrée dans les articles lus où les auteurs se hâtent à pré-traiter pour calculer, nous semble une étape importante de prise de contact avec ce dernier. Ces premières analyses informent aussi bien sur le type de tokens que nous devront manipuler que sur leur distribution. Cette étape permet alors de relever le bruit que certains tokens pourraient apporter. Cette introduction au corpus permet également d'éclairer quant aux choix qui pourraient être applicables lors d'un preprocessing. Les nuages de mots pour leur part nous renseignent en laissant émerger thématiques et mots clés que nous devrions retrouver par le biais des calculs.

CHAPITRE 4

RÉSULTATS

Sommaire

4.1 Introduction

L'idée est de poser les différentes briques dans un but final d'effectuer une analyse de sentiments d'un véritable corpus. Nous supposons que les problèmes, les questions soulevés à l'échelle de ce petit corpus apporteront des réponses pour une version finale. Cependant il est évident que la taille du corpus et une échelle temporelle plus étendue permettent de filtrer les topics de part leur fréquence et leur présence dans un continuum temporel qui éliminera naturellement les singularités.

L'annotation est une tâche non seulement ardue mais aussi longue. C'est un véritable goulot d'étranglement. Peu appréciée, souvent dénigrée et minimisée elle s'avère pourtant indispensable dans la mise en place de méthodes supervisées. L'annotation nécessite la rédaction d'un guide qui compile les règles décrivant les concepts à annoter et les formes linguistiques qu'ils prennent dans le texte[1]. Le processus est itératif et ajusté au fil des premières expériences d'annotation. Des métriques telles que le *Kappa de Cohen* ou le *Kappa de Fleiss* permettent d'évaluer

1. cf. Cours de Mme Gianola *Introduction aux principes de l'annotation manuelle*, *Le guide d'annotation* & *Accord inter-annotateur*

l'accord inter-annotateur (*AIA*) autrement dit de comparer les annotations produites par des annotateurs. Les données annotées sont des ressources essentielles pour les activités *TAL*. Elles servent notamment d'exemple pour entraîner les systèmes d'apprentissage automatique. Cependant, nous avons fait le choix de découvrir, d'élaborer et de tester des modèles pour cette étude qui ne seront que plus efficaces combinés à un corpus qui sera annoté dans le futur (cf. le guide d'annotation du SemEval 2016 Task 5 *Aspect Based Sentiment Analysis (ABSA-16)* [2, 3]). Nous partons donc d'un corpus brut tiré des réseaux sociaux et non annoté. Aucun transfert de variables à l'instar des évaluations étoilées n'est possible, il aurait dès lors constitué une annotation.

La première partie de ce chapitre consiste à tester des librairies et des méthodes qui pourraient répondre à la première tâche qui nous intéresse à savoir l'analyse de sentiments à l'échelle du document. Une fois cette étape réalisée, il devient possible d'entrainer et de construire un modèle qui permettra de prédire le sentiment des tweets. La deuxième partie de ce chapitre se place à une échelle plus fine : celle des aspects. Une première étude s'intéresse à différentes techniques de *Topic Modeling* que nous avons testés et que nous présentons. Une seconde étude entamée propose des pistes pour associer aux aspects la polarité à laquelle ils correspondent.

4.2 TextBlob et Vader pour l'analyse de sentiments

4.2.1 TextBlob

Textblob (cf. Section 1.3) fournit deux résultats : la polarité et la subjectivité. Sa polarité se situe dans l'intervalle [-1,1], -1 définissant un sentiment très négatif, 1 un sentiment très positif et 0 un sentiment neutre. Dans la littérature le sentiment neutre est associé à 0 uniquement, nous avons introduit et testé un seuil de 0.05 pour être moins stricte et voir quel était l'impact sur les résultats.

Les Figures 4.1 et 4.3 fournissent la distribution des polarités sur un corpus brut : le sentiment est majoritairement positif à hauteur de 44%. Le sentiment neutre est élevé : il atteint les 39%, le sentiment négatif est minoritaire avec 16%. Le corpus a ensuite été prétraité légèrement (de façon « soft ») (emojis remplacés par le texte [4] correspondant en utilisant la librairie Advertools, certaines ponctuations enlevées, les URLs remplacées par *url*). Les résultats sont fournis par la Figure 4.4 : il n'y a quasi pas de différence dans le comportement. Une dernières expériences a constitué à introduire un seuil de 0.05 pour délimiter un intervalle de tweets étiquetés neutres et non une valeure unique (0). La Figure 4.2 montre une répartition revue des polarités : le neutre devient majoritaire à hauteur de 50%, les étiquettes positives perdent 2%, les étiquettes négatives diminuent de 3%.

N.B. : Le code utilisé est saTextblobVader.py.

2. https://alt.qcri.org/semeval2016/task5/data/uploads/absa2016_annotationguidelines.pdf

3. https://alt.qcri.org/semeval2016/task5/

4. *Certaines publications proposées de remplacer les émojis par le texte correspondant, nous avons donc testé. Expériences « emo »*

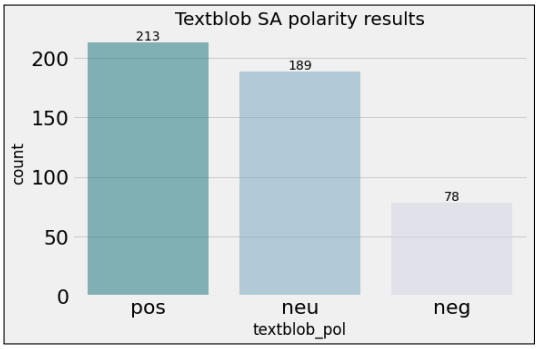


FIGURE 4.1 – Distribution des polarités avec Textblob.

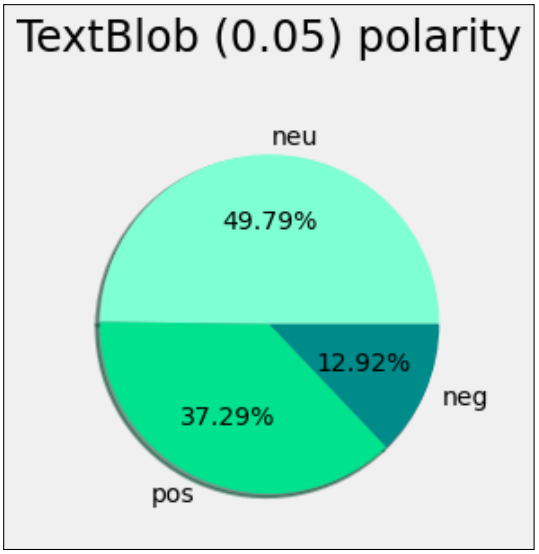


FIGURE 4.2 – Polarité avec Textblob (seuil=0.05).

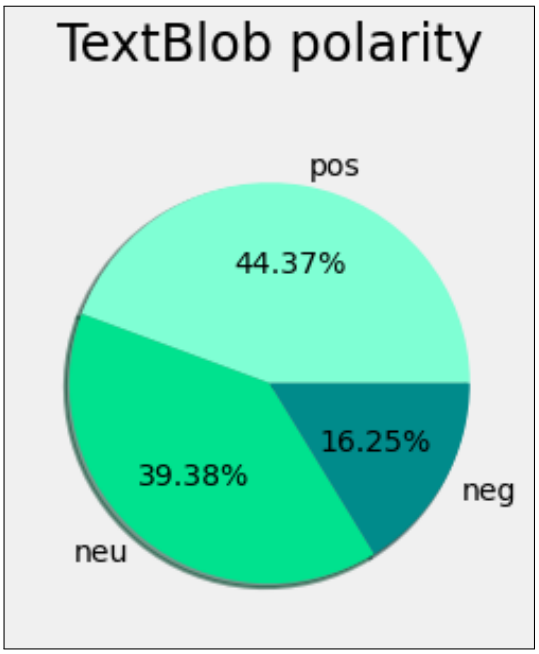


FIGURE 4.3 – Polarité Textblob sur corpus brut.

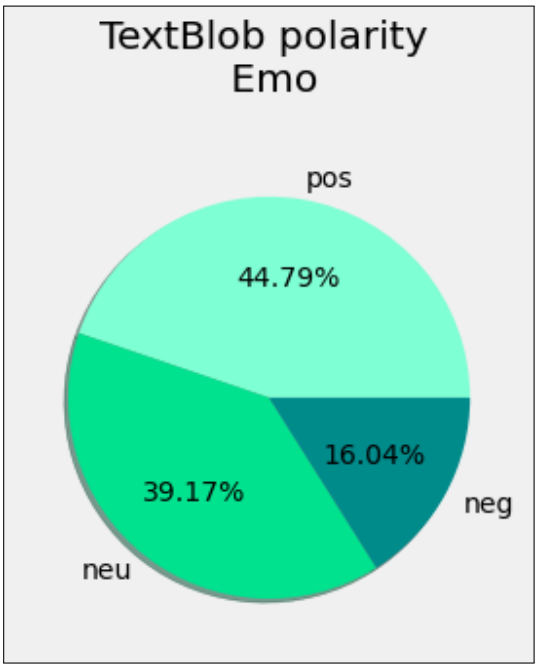


FIGURE 4.4 – Polarité Textblob sur corpus avec un preprocessing *soft*.

4.2.2 VADER

VADER (cf. Section 1.4) fournit quatre résultats : les probabilités d'être positif/-négatif/neutre et un score de l'intensité du sentiment *compound*. VADER interprète les émojis, les symboles # et @ rendent l'évaluation du token neutre. L'analyseur est sensible aux ponctuations, à la casse comme on peut le constater sur la Figure 4.5.

```
analyzer.polarity_scores("This is great")
{'neg': 0.0, 'neu': 0.328, 'pos': 0.672, 'compound': 0.6249}

analyzer.polarity_scores("This is GREAT")
{'neg': 0.0, 'neu': 0.293, 'pos': 0.707, 'compound': 0.7034}

analyzer.polarity_scores("This is GREAT!!!")
{'neg': 0.0, 'neu': 0.259, 'pos': 0.741, 'compound': 0.7723}

emojisList=["<3", ":)", ":p",":("]

for tweet in emojisList:
    print(analyzer.polarity_scores(tweet))
{'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound': 0.4404}
{'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound': 0.4588}
{'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound': 0.34}
{'neg': 1.0, 'neu': 0.0, 'pos': 0.0, 'compound': -0.4404}

withAndWithoutHash=["#good", "good", "@good"]

for tweet in withAndWithoutHash:
    print(analyzer.polarity_scores(tweet))
{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0}
{'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound': 0.4404}
{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0}

withAndWithoutHash=["#arcelormittal", "arcelormittal", "@arcelormittal"]
for tweet in withAndWithoutHash:
    print(analyzer.polarity_scores(tweet))
{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0}
{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0}
{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0}
```

FIGURE 4.5 – Quelques tests basiques de VADER.

Les expériences faites sont les mêmes que pour Textblob. Les Figures 4.6 et 4.7 révèlent une polarité majoritairement positive avec 50%. Le sentiment neutre atteint 34%, la polarité négative est de 16%. L'expérience menée sur le corpus prétraité voit les scores positif et négatif augmenter très légèrement et le score neutre diminuer d'un point cf. Figure 4.8.

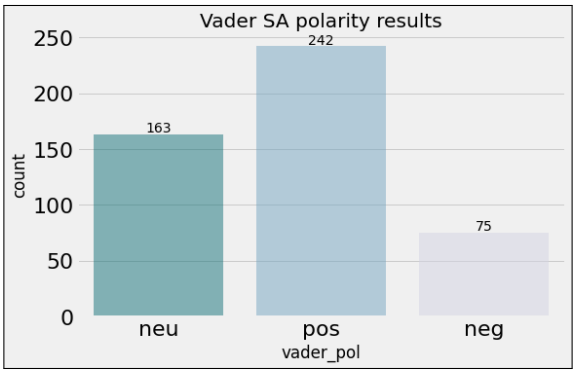


FIGURE 4.6 – Polarité avec VADER.

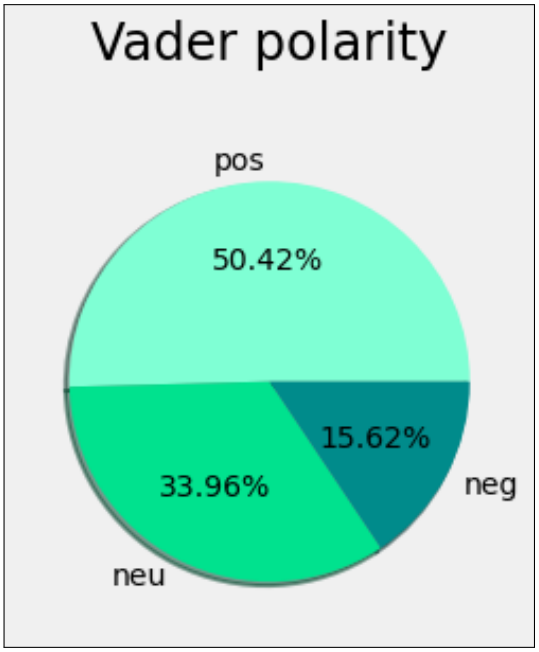


FIGURE 4.7 – Polarité Vader sur corpus brut.

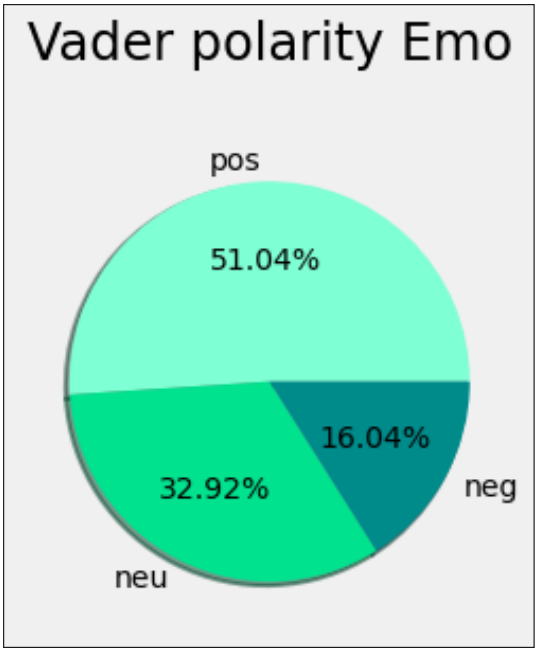


FIGURE 4.8 – Polarité Vader sur corpus avec un preprocessing *s*oft.

4.2.3 Comparaison des résultats de TextBlob et de Vader

Les Graphes 4.1 et 4.6 montrent un comportement général similaire : majorité positive, puis neutre et enfin négative. L'écart se creuse entre le nombre de positifs et de neutres néanmoins. Les Figures 4.10 et 4.11 présentent le nombre de tweets évalués identiques : seule la moitié du corpus est étiquetée par les modèles de la même façon (239 tweets sur 480 voire 240 sur 480). Le Graphe 4.9 donne la distribution des tweets dont le label est commun. Les expériences montrent que l'orientation du sentiment n'est pas distribuée de la même façon par ces modèles. L'article [László and Attila, 2021] souligne ce même constat suite à la comparaison d'un RNN construit et de Textblob.

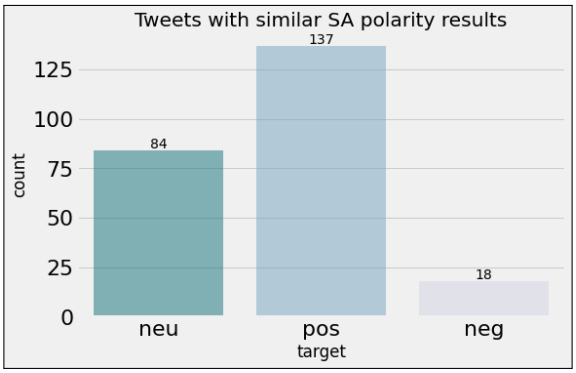


FIGURE 4.9 – Polarité similaire entre VADER et Textblob.

Que se passe-t-il à l'échelle du texte ? Dans la Figure 4.12 le premier exemple montre que les modèles ont mal évalué le texte : la réduction d'émission de CO2 est étiquetée *neutre*. Le deuxième exemple est étiqueté par VADER *neutre* tandis que

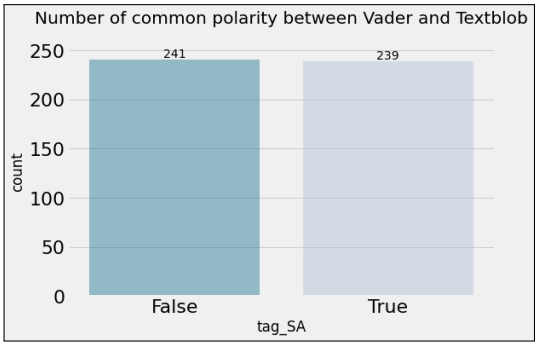


FIGURE 4.10 – Distribution des polarités communes sur corpus brut.

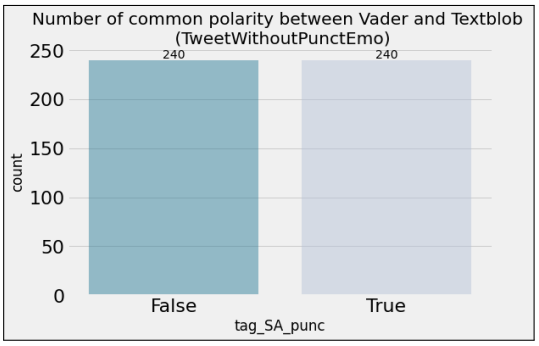


FIGURE 4.11 – Distribution des polarités communes sur corpus avec un preprocessing *soft*.

Textblob l'évalue positivement ce qui est effectivement le cas. Les modèles ne doivent pas couvrir le champ lexical.

Les textes de la Figure 4.13, pourtant simples et courts, ont été étiquetés de la même façon par contre l'orientation n'est pas correcte.

En conclusion, l'idée initiale d'accorder une certaine confiance à l'étiquetage des tweets évalués de la même façon par les deux modèles échoue. Nous nous dirigeons par conséquent vers des modèles de Deep Learning de type BERT, modèles pré-entrainés.

FIGURE 4.12 – Quelques expériences VADER & Textblob.

FIGURE 4.13 – Quelques expériences VADER & Textblob.

4.3 BERT pour l'analyse de sentiments

4.3.1 NLPTown : bert-base-multilingual-uncased-sentiment

Ce modèle a été développé par l'entreprise *NLP Town*. Il a été entrainé sur des commentaires d'évaluation de produits et prédit le sentiment d'un commentaire à l'instar d'une échelle de notations allant de 1 à 5. Il est destiné à être utilisé comme modèle d'analyse pour l'évaluation de produits ou pour d'autres tâches d'analyse de sentiments. Ainsi, même si la nature de notre corpus est éloigné de part le contenu mais aussi de part la structure syntaxique des tweets : nous pouvons le tester.

Un prétraitement très léger a été appliqué au corpus brut : suppression des urls et des mentions d'utilisateur. L'instanciation du modèle a été effectuée en utilisant le modèle pré-entrainé suivant *nlptown/bert-base-multilingual-uncased-sentiment*. Les tweets sont alors ensuite encodés, puis le sentiment calculé (cf. Figure 4.15).

La tendance générale a évolué en comparaison aux résultats fournis par Vader et Textblob : majoritairement le sentiment reste positif à hauteur de 49%, le sentiment négatif a doublé atteignant les 31%, le sentiment neutre a diminué de moitié s'élevant à 20% si l'on efface les nuances (cf. Figure 4.14).

Qu'en est-il de ces résultats ? Sont-ils fiables ? Une vérification humaine et manuelle a permis de tirer comme première conclusion que l'affectation du sentiment par ce modèle était meilleure que celle de Vader ou de Textblob. Néanmoins, une seconde conclusion est que les résultats peuvent cependant ne pas être corrects non plus. Le cas du tweet de la Figure 4.16b montre une très mauvaise notation : le sentiment étiqueté est très négatif là où il devrait être a minima positif.

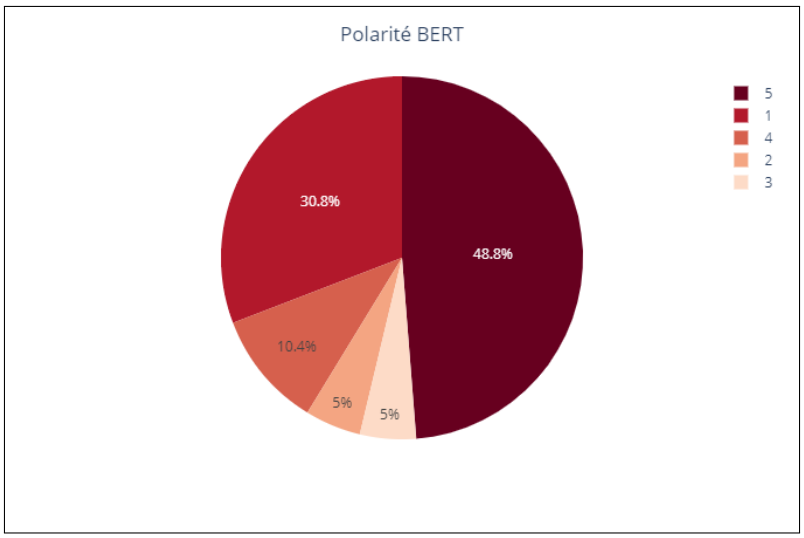


FIGURE 4.14 – Polarité BERT sur corpus avec preprocessing *s*oft.

```
df_without_RT[['sentimentBert','softclean_tweet']].head(10)
```

	sentimentBert	softclean_tweet
0	5	ArcelorMittal Has World's First 'Green' Steel ...
1	5	'A 50/50 joint venture between ArcelorMittal ...
2	5	ArcelorMittal Investing Nearly $400 Million in...
3	5	ArcelorMittal Has World's First 'Green' Steel ...
4	1	$MT ArcelorMittal call volume above normal and...
5	1	$MT ArcelorMittal call volume above normal and...
6	5	"It is awesome! It was wonderful experience an...
7	2	The steel industry continues to negotiate with...
8	5	Check out how ArcelorMittal is improving effic...
9	5	ArcelorMittal, the world's largest steel produ...

FIGURE 4.15 – Tweets affectés d'une polarité.

```
(df_without_RT['softclean_tweet'].iloc[2],df_without_RT['sentimentBert'].iloc[2])
```

Python

```
('ArcelorMittal Investing Nearly $400 Million in Eliminating Carbon Emissions  via ',
 5)
```

(a) Tweet noté positivement

```
(df_without_RT['softclean_tweet'].iloc[472],df_without_RT['sentimentBert'].iloc[472])
```

Python

```
("Last week, the world's second largest steel maker Arcelormittal launched a green steel carbon
offset program, and announced a plan to invest $100M a year in green steel initiatives.  ",
 1)
```

(b) Tweet noté négativement

FIGURE 4.16 – Tweets et leur sentiment

4.3.2 TweetEval : Twitter-roBERTa-base-sentiment

Le modèle *Twitter-roBERTa-base-sentiment* est basé sur roBERTa. Il a été entrainé sur environ 58 millions de tweets et a été ajusté pour l'analyse de sentiment sur le benchmark *TweetEval* [5]. TweetEval se compose de sept tâches de classifications hétérogènes sur Twitter en langue anglaise. Ces tâches sont l'analyse de sentiment, la reconnaissance des émotions, la détection de langage offensant, la détection de discours haineux, la prédiction de position, la prédiction d'emoji et la détection d'ironie [Barbieri et al., 2020]. Chaque ensemble a été élaboré en suivant le même format ainsi si l'on souhaite par exemple s'intéresser aux émotions il suffit de remplacer dans le modèle suivant *MODEL = f"cardiffnlp/twitter-roberta-base-task" task* par le type de tâche *task='emotion'*. L'ensemble des valeurs que *task* peut prendre sont : *sentiment, emotion, irony, hate, offensive, stance, emoji* et *emotion*.

N.B. : Seuls les tweets ayant plus de 3 tokens et sans URLs ont été pris en compte dans TweetEval de sorte à éviter les tweets automatiques et les spams.

Le préprocessing mis en place a consisté à remplacer les URLs par *http* et les mentions d'utilisateur par *@user*. De la même façon, la mise en œuvre passe par l'instanciation du modèle. Le texte, une fois encodé, est alors fourni au modèle et la tâche d'analyse de sentiment appliquée (cf. Figure 4.18). Les différentes étiquettes de classification du modèle sont : *positive, negative* et *neutral*. Ce modèle, entrainé sur des tweets, modifie la tendance générale qui jusque là était identique modulo une distribution différente des documents dans les trois classes. Cette fois la classe *neutre* s'élève à plus de 65% en comptant 316 tweets sur 480. Le sentiment positif reste toujours plus important que le sentiment négatif avec respectivement $\sim$26% et $\sim$9% représentant d'une part 123 tweets et d'autre part 41 tweets (cf. Figure 4.17).

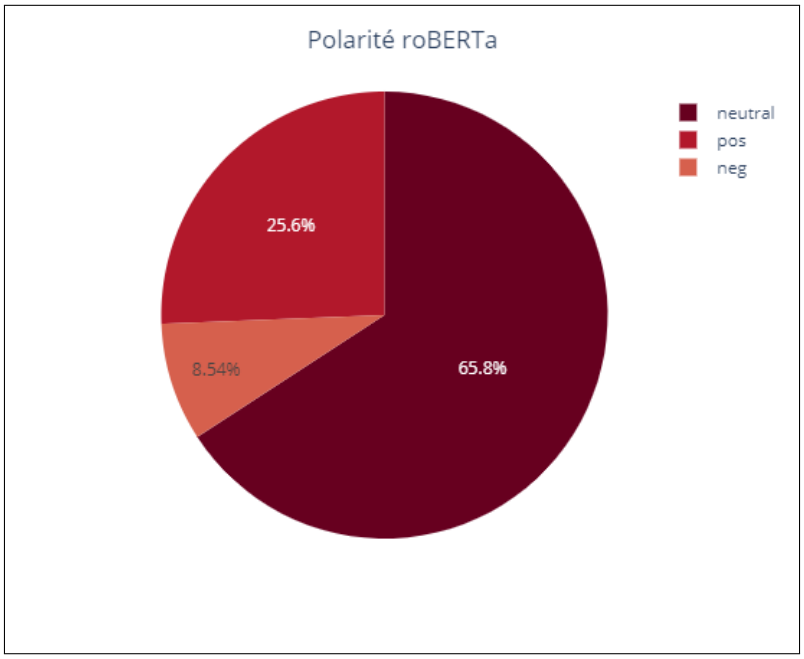


FIGURE 4.17 – Polarité de Twitter-roBERTa sur corpus avec preprocessing *s*oft.

5. Dépôt officiel de *TweetEval* https://github.com/cardiffnlp/tweeteval

```
df_without_RT[['twitterRobertaSentiment','pre_full_tweet']]
```
Python

	twitterRobertaSentiment	pre_full_tweet
0	{'positive': 0.2062, 'negative': 0.0082, 'neut...	ArcelorMittal Has World's First 'Green' Steel ...
1	{'positive': 0.0334, 'negative': 0.023, 'neutr...	@user 'A 50/50 joint venture between ArcelorMi...
2	{'positive': 0.2358, 'negative': 0.0099, 'neut...	ArcelorMittal Investing Nearly $400 Million in...
3	{'positive': 0.1824, 'negative': 0.0139, 'neut...	ArcelorMittal Has World's First 'Green' Steel ...
4	{'positive': 0.6545, 'negative': 0.0028, 'neut...	$MT ArcelorMittal call volume above normal and...
...	...	...
475	{'positive': 0.6661, 'negative': 0.0026, 'neut...	In case you missed it: ArcelorMittal's low-car...
476	{'positive': 0.0653, 'negative': 0.0388, 'neut...	September 2019: Lex Greensill, a confidant of ...
477	{'positive': 0.1292, 'negative': 0.1715, 'neut...	@user @user @user @user @user Reminds me of th...
478	{'positive': 0.0499, 'negative': 0.4939, 'neut...	@user Sir i belong to a poor family\ni want to...
479	{'positive': 0.6341, 'negative': 0.0042, 'neut...	@user committed to the development and rollout...

480 rows × 2 columns

FIGURE 4.18 – Tweets affectés d'une polarité Twitter-roBERTa.

Aucun des modèles jusqu'à présent testés ne fournit exactement les mêmes résultats. Il est nécessaire que l'humain vérifie, valide ou rejette ces derniers. Si nous nous intéressons aux évaluations fournies : elles semblent être dans l'ensemble correctes voire meilleures. Prenons les tweets de la Figure 4.19 : le *tweet 472* qui avait été noté par *bert-base-multilingual-uncased-sentiment* comme étant très négatif est noté par *Twitter-roBERTa* comme étant positif, ce qui est effectivement le cas. Le *tweet 24* est correctement noté négativement tandis que le *tweet 2* étiqueté neutre aurait dû être étiqueté positif ce que *bert-sentiment* avait correctement noté.

Le temps manquant nous n'avons pas pu aborder l'étape suivante sur ces deux derniers modèles à savoir utiliser les résultats obtenus comme données d'entrée permettant la construction d'un modèle de classification permettant alors de prédire le sentiment des tweets.

N.B : D'autres modèles ont été ciblés et peuvent être testés en particulier *DistilBERT-tweet-eval-emotion* [6] et *Bertweet* [7].

6. https://huggingface.co/philschmid/BERT-tweet-eval-emotion
7. https://huggingface.co/transformers/model_doc/bertweet.html

```
(df_sent['softclean_tweet'][2],df_sent['sent'][2])
```
Python

```
('ArcelorMittal Investing Nearly $400 Million in Eliminating Carbon
Emissions  via ',
 'neutral')
```

(a) Tweet noté neutre

```
(df_sent['softclean_tweet'][472],df_sent['sent'][472])
```
Python

```
("Last week, the world's second largest steel maker Arcelormittal
launched a green steel carbon offset program, and announced a plan to
invest $100M a year in green steel initiatives.  ",
 'positive')
```

(b) Tweet noté positivement

```
(df_sent['softclean_tweet'][24],df_sent['sent'][24])
```
Python

```
('I love the dunes but I hate the steel mills so much. Might have to
go Ted K mode on arcelor mittal mfs dump chemicals like chromium and
ammonium/cyanide in the the lake every year + they emit like 18,000
pounds of lead into the air btw',
 'negative')
```

(c) Tweet noté négativement

FIGURE 4.19 – Tweets et leur *Twitter-roBERTa* sentiment

4.4 L'extraction d'aspects (ATE & ABSA)

Certaines thématiques majeures sont d'ores et déjà ciblées parmi elles : Innovation, Sustainability, Environment, CO2, Additive manufacturing, Digital. Quelques filiales également notamment XCarb, InduSteel, MSNIndia, et AMNSCalvert.

4.4.1 Topic Modeling avec LDA

Le code correspondant est *ldaTopicVF.ipynb*.

Le texte a été prétraité de sorte à ne pas être perturbé par des élements tels que ponctuations, users mention, urls, stopwords ou emojis. La taille des tweets préprocessés varie de 1 à 83 tokens. On choisit alors de lemmatiser les tweets tout en conservant et en supprimant la casse. Celà a un impact sur l'étiquettage de spaCy qui considère souvent les token commençant par une majuscule comme un pronom.

Par exemple "Green" est étiquetté comme étant un pronom tandis que "green" est étiquetté comme étant adjectif cf. Figure 4.51..

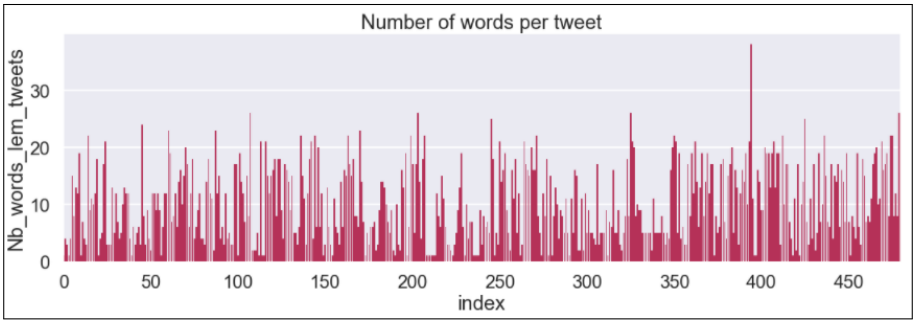


FIGURE 4.20 – Distribution du nombre de mots par tweet - Corpus avec preprocessing.

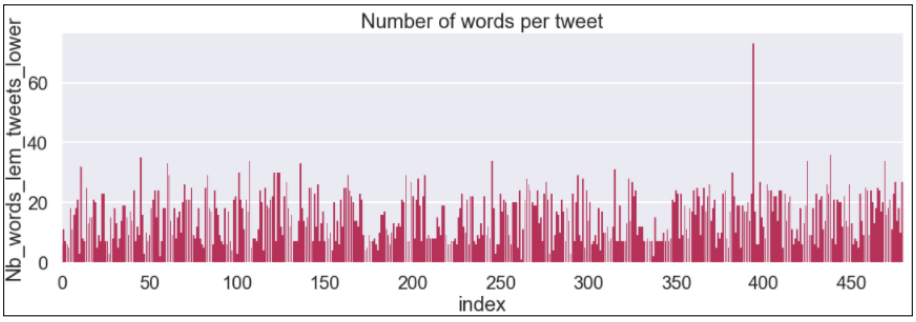


FIGURE 4.21 – Distribution du nombre de mots par tweet - Corpus avec preprocessing + lowercase.

Dans le cas du corpus lemmatisé avec ou sans majuscule, la taille des tweets varie de 1 à 38 mots et de 1 à 73 mots (cf. les figures 4.20 et 4.21) respectivement suite à un filtrage à travers spaCy des parties du discours suivants : nom, verbe, adjectif et adverbe. Nous partons de ces deux ensembles pour constituer les dictionnaires de vocabulaire. Nous utilisons la librairie gensim et importons le module corpora. Une fois le dictionnaire créé, on crée la matrice document/termes. Chaque ligne correspond à un document ici un tweet et chaque colonne correspond à un mot. La matrice est de dimension : NbDoc x VocabSize.

On crée alors le modèle. Les paramètres utilisés (voir tableau 4.1) :

On constate que les thématiques trouvés sont assez intéressantes, la fréquence d'appartenance des termes aux topics semble correct. Il semblerait que cette voie puisse mener à l'élaboration de topics il reste cependant un travail de tuning mais aussi de filtrage à mettre au point. Aussi la taille du corpus n'aide pas. Un corpus plus volumineux mais aussi qui s'étend sur un continuum temporel permettrait naturellement de filtrer les bruits liés à une situation locale et singulière.

Malgré celà nous proposons d'aller un peu plus loin afin de mettre en place les outils qui pourront servir à moyen terme afin de continuer cette étude.

Deux grandeurs sont disponibles pour ce faire : *perplexité* et le *score de cohérence*. Le nombre de topics peut être choisit grâce à un *score de cohérence*

Pour les modèles que nous avons entrainé, les valeurs de perpléxité sont de -7.01 et -7.11 respectivement, pour les scores de cohérence : 1.44 et 0.41 respectivement.

Paramètres	LDA	Définitions
corpus	doc_term_matrix	Matrice document/terme
id2word	dictionnary	Mapping ID des mots/mots
num_topics	10	Nombre de topics à extraire du corpus
chuncksize	50	Nombre de documents à utiliser pour chaque entrainement de sous ensemble.
passes	50	Nombre de fois que l'on parcourt le corpus
iterations	100	Nombre maximum d'itérations à travers le corpus lors de la génération des topics.

TABLE 4.1 – Paramètres pour le modèle LDA

Le choix du nombre de topics peut être, théoriquement, effectué en considérant la valeur du score de cohérence en fonction du nombre de topics. Des calculs ont par conséquent été lancés en faisant varié le nombre de thématiques de 2 à 50. Il s'agit alors de sélectionner le nombre de topics qui correspond au meilleur score de cohérence. Cependant plus le nombre de topics est important plus la probabilité de répétitions de mots-clé est probable. Les graphes suivants 4.27 et 4.28 représentent la variation du score de cohérence en fonction des topics. On constate en particulier dans le graphe 4.28 qu'il n'est pas aisé de conclure instinctivement. Nous proposons de montrer quelques résultats suite aux choix suivants : un nombre de topics de 15 et de 28 respectivement cf. les figures 4.30 et 4.31. Il s'avère très difficile de tirer profit de ces ensembles.

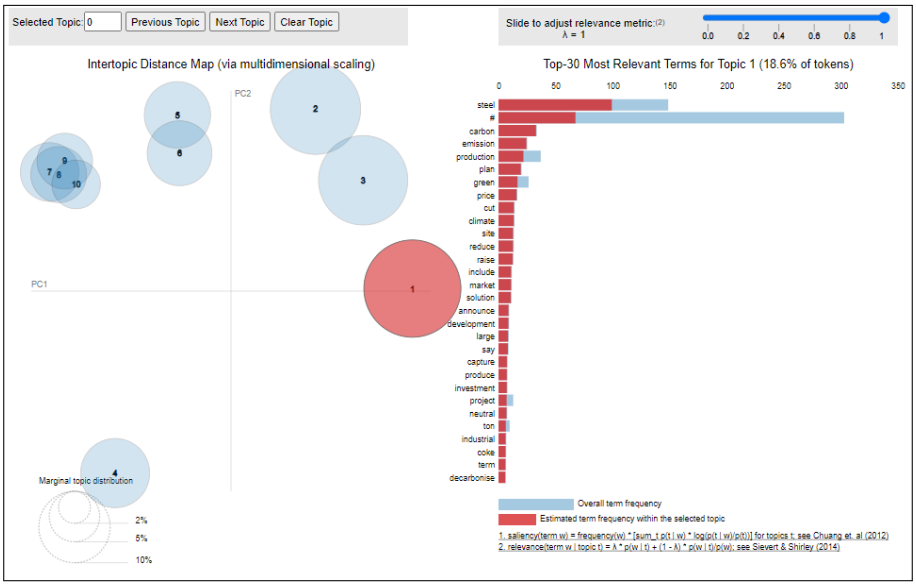


FIGURE 4.22 – Topic 1 relatif au *Carbon* - Corpus avec preprocessing.

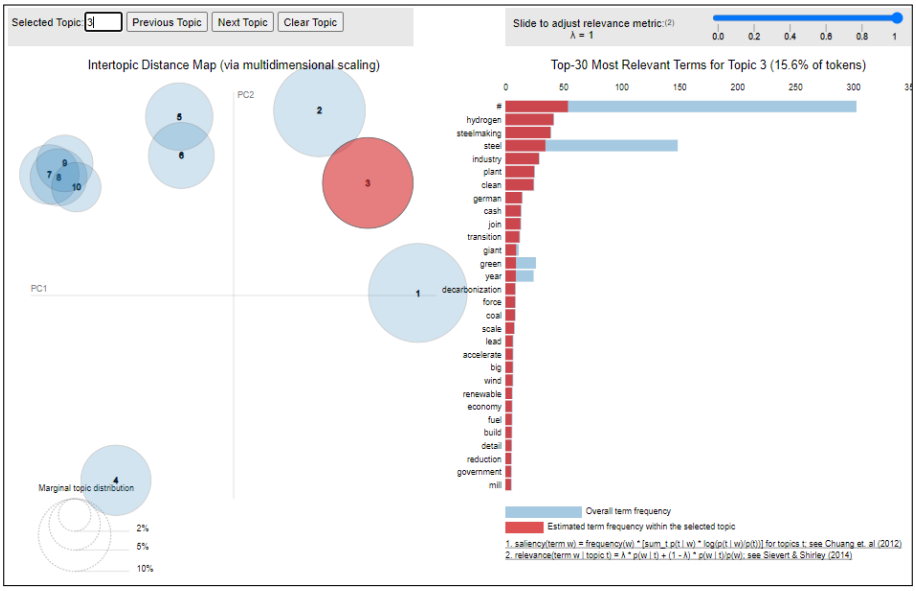


FIGURE 4.23 – Topic 3 relatif à *Green* - Corpus avec preprocessing.

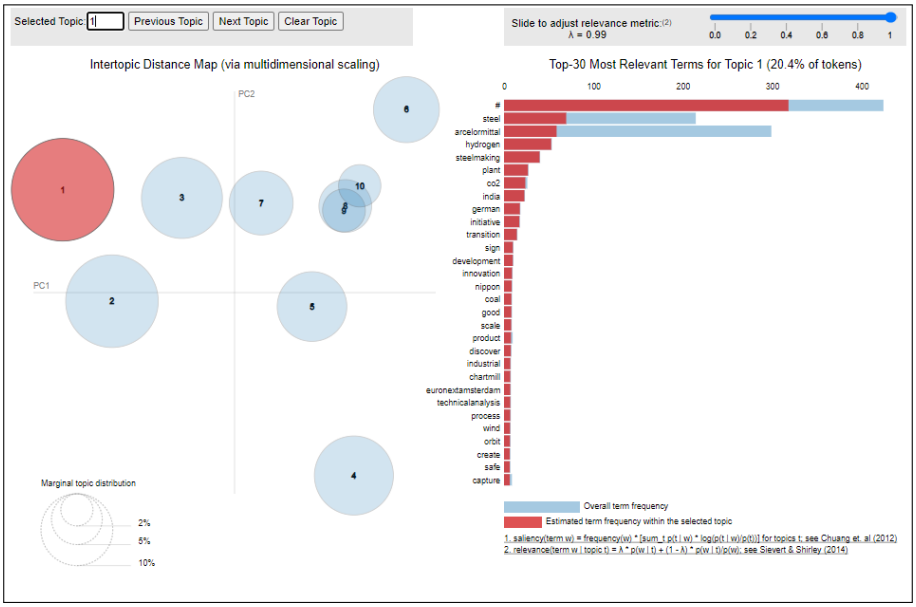


FIGURE 4.24 – Topic 1 relatif au *C02* - Corpus avec preprocessing + lowercase.

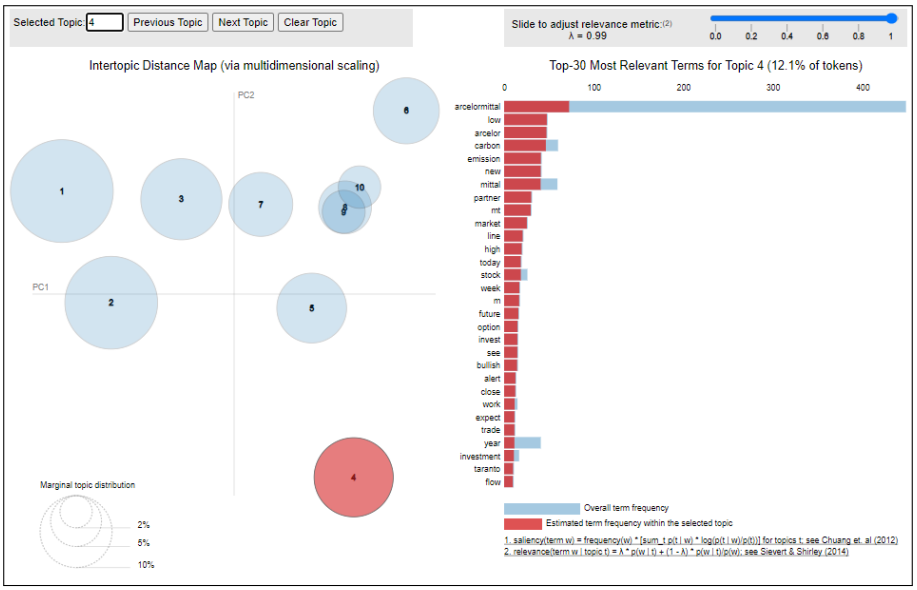


FIGURE 4.25 – Topic 4 relatif au *Carbon* - Corpus avec preprocessing + lowercase.

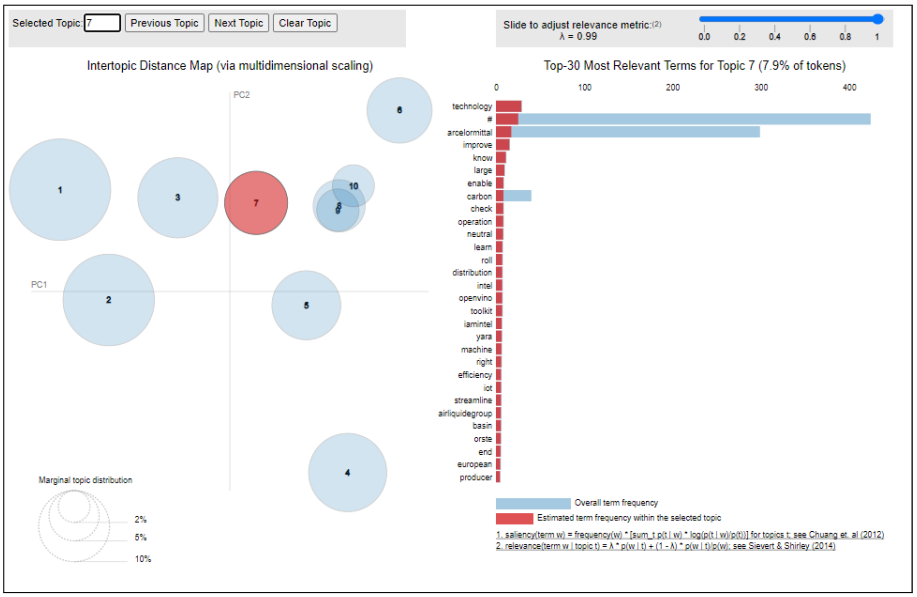


FIGURE 4.26 – Topic 7 relatif à la *Technoligie* - Corpus avec preprocessing + lowercase.

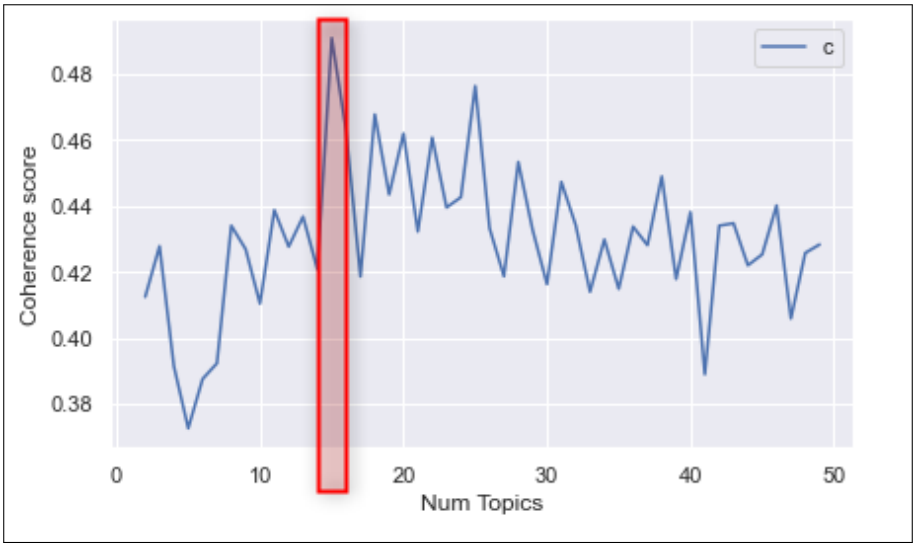


FIGURE 4.27 – Score de cohérence en fonction du nombre de topics.

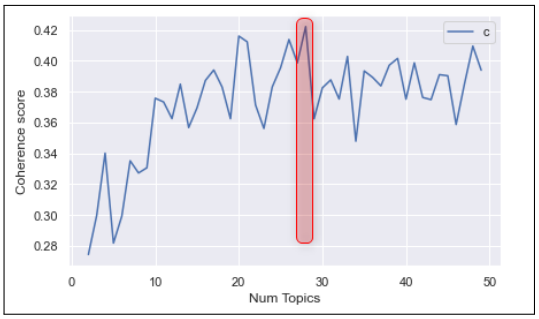


FIGURE 4.28 – Score de cohérence en fonction du nombre de topics (corpus en lowercase).

```
# Print the coherence scores
x = range(2, limit, 1)
for m, cv in zip(x, coherence_values_lower):
    print("Num Topics =", m, " has Coherence Value of", round(cv, 4))
```

```
Num Topics = 2  has Coherence Value of 0.2744
Num Topics = 3  has Coherence Value of 0.3
Num Topics = 4  has Coherence Value of 0.3402
Num Topics = 5  has Coherence Value of 0.2818
Num Topics = 6  has Coherence Value of 0.2993
Num Topics = 7  has Coherence Value of 0.3352
Num Topics = 8  has Coherence Value of 0.3273
Num Topics = 9  has Coherence Value of 0.3307
Num Topics = 10  has Coherence Value of 0.3758
Num Topics = 11  has Coherence Value of 0.3733
Num Topics = 12  has Coherence Value of 0.3626
Num Topics = 13  has Coherence Value of 0.3849
Num Topics = 14  has Coherence Value of 0.3568
Num Topics = 15  has Coherence Value of 0.3696
Num Topics = 16  has Coherence Value of 0.3873
Num Topics = 17  has Coherence Value of 0.3941
Num Topics = 18  has Coherence Value of 0.383
Num Topics = 19  has Coherence Value of 0.3625
Num Topics = 20  has Coherence Value of 0.4161
Num Topics = 21  has Coherence Value of 0.4124
Num Topics = 22  has Coherence Value of 0.3714
Num Topics = 23  has Coherence Value of 0.3562
Num Topics = 24  has Coherence Value of 0.3831
Num Topics = 25  has Coherence Value of 0.3956
Num Topics = 26  has Coherence Value of 0.4139
Num Topics = 27  has Coherence Value of 0.3987
Num Topics = 28  has Coherence Value of 0.4222
Num Topics = 29  has Coherence Value of 0.3625
Num Topics = 30  has Coherence Value of 0.3825
Num Topics = 31  has Coherence Value of 0.3877
Num Topics = 32  has Coherence Value of 0.3753
Num Topics = 33  has Coherence Value of 0.4029
Num Topics = 34  has Coherence Value of 0.3479
Num Topics = 35  has Coherence Value of 0.3935
```

FIGURE 4.29 – Affichage du score de cohérence en fonction du nombre de topics (corpus en lowercase).

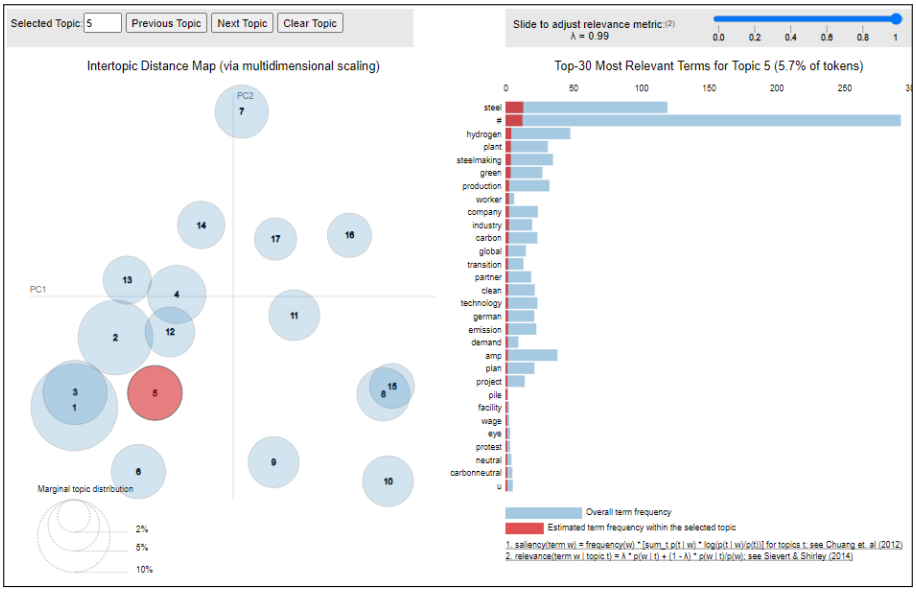


FIGURE 4.30 – Exemple de topic suite à optimisation du score de cohérence.

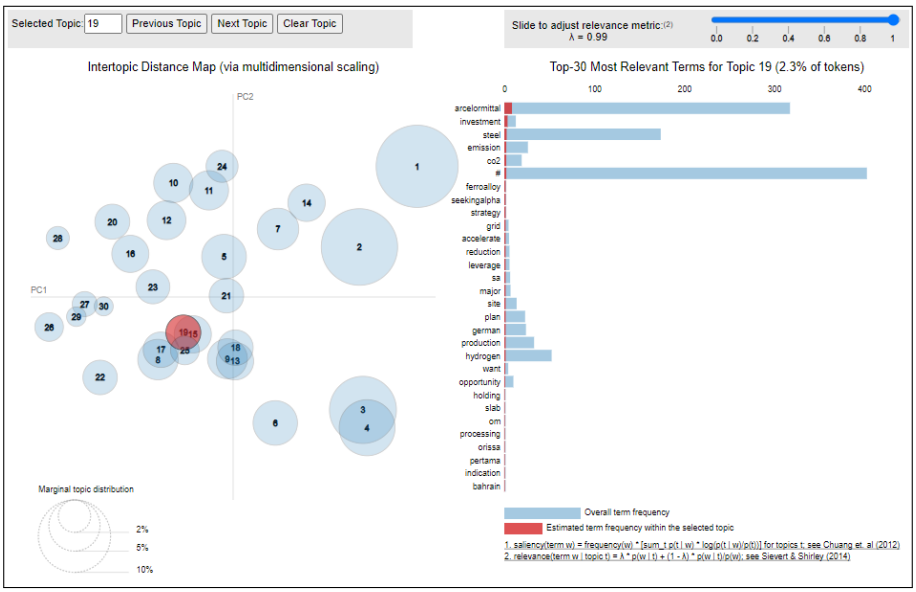


FIGURE 4.31 – Exemple de topic suite à optimisation du score de cohérence (corpus en lowercase).

L'étude doit être poussée afin de visualiser les mots clés ainsi que leur poids, leur fréquence d'apparition pour chaque topic.. il faut aussi filtrer parmis les mots qui émergent ceux qui sont les plus pertinents. Nous pouvons dire que la LDA peut être une étape dans la construction d'un pipeline qui répond au sujet d'analyser les sentiments en étant plus fin que l'échelle globale d'un document ou d'une phrase.

Nous proposons de poursuivre sur le chemin de la découverte d'autres méthodes afin d'avoir des points de comparaison et une vision plus large de ce qu'il est possible de mettre en place.

4.4.2 Topic Modeling avec des modèles pré-entrainés de type Bert

Nous proposons de présenter plusieurs modèles pré-entrainés qui ont donné des résultats intéressants. *Hugging Face*, pointe de la technologie en NLP, a pour cela été d'une grande aide. Leur site web [8] fournit une multitude de modèles basés sur les *Transformers* facilitant la mise en œuvre de l'entrainement des modèles en quelques lignes de code aussi bien pour la compréhension du language naturel (*NLU - Natural Language Understanding*) que pour la génération du langage naturel (*NLG - Natural Languge Generation*) avec plus de 32 modèles pré-entrainés supportant plus de 100 langues ainsi qu'une interopérabilité entre notamment *PyTorch* et *TensorFlow*.

BERTopic

BERTopic est une technique de topic modeling qui exploite les intégrations *BERT* et le *c-TF-IDF (class-based TF-IDF)* pour créer des clusters denses facilement interprétables tout en conservant les mots importants dans les descriptions des sujets [Grootendorst, 2020a].

Pour le prétraitement : plusieurs expériences ont été effectuées. Le prétraitement concerne : les *url*, les *"user mention"*, les *ponctuations*, les *stopwords*, les *digits "purs"* et la *casse*. Il s'avère que le modèle est très sensible au prétraitement même si celui-ci est assez lisse, *soft* (nous n'utilisons ni la radicalisation ni la lemmatisation, etc.). Une première conclusion est qu'il n'est donc pas possible de prendre un modèle et d'appliquer brutalement un pré-traitement sans une étude préalable de la sensibilité vis à vis de ce dernier. Dans notre cas la taille du corpus n'aide évidemment pas cependant l'idée générale reste identique. Les différentes possibilités sont données dans le Tableau 4.2 : les url sont soit remplacées par *"url"* soit enlevées, les user mention sont soit remplacées par *"usermention"* soit enlevées, les ponctuations sont toutes supprimées ou toutes supprimées excepté le point, la virgule, le point d'exclamation et le point d'interrogation. Les digits "purs" sont enlevés. La présence et l'absence des stopwords a aussi été évalué de la même façon que la sensibilité à la casse. A noter qu'un espace a aussi été inséré entre les ponctuations et les mots de sorte que les mots soient trouvés dans les dictionnaires et de sorte à ne pas polluer le vocabulaire par une ponctuation accolée.
Ainsi, des expériences peuvent être menées afin de constater les conséquences sur le résultat final. Nous ne présentons pas tous les résultats mais certains qui nous semblent pertinents et qui pourraient être creusés dans le futur.

Le code correspondant à cette section (évaluation de BERTopic) est le notebook suivant *bertTopicVF.ipynb* ainsi que le module de préprocessing *mod_preprocessing.py*.

8. https://huggingface.co/transformers/index.html#

N.B. : Quelques incompatibilités entre les versions numpy et BERTopic peuvent apparaître : il suffit d'upgrader numpy de redémarrer le notebook pour que le problème soit résolu.

Le prétraitement, le tuning et l'étude de la sensibilité nécessitent divers allers-retours avant d'arriver à la sélection de clusters optimaux, clusters de thématiques appelés aussi *topics*. Ces derniers embarquent un ensemble de *topic-words*, que l'on pourrait traduire de *mots-clés*. Il s'agit d'ensembles de mots les plus représentatifs des clusters formés.

La mise en œuvre est par ailleurs simple et efficace grâce à l'utilisation des *Transformers* de *Hugging Face* et aux outils performants et puissants de visualisation proposés. Pour la prise en main *Hugging Face*, sa stratégie open-source, ses différents articles et son *GitHub* sont des outils remarquables qui aident à monter en compétences de façon très efficace. On peut se référer notamment aux articles de *Towards Data Science* [9, 10].

Un modèle BERTopic a été créé avec activation du calcul des probabilités : elle permet notamment d'extraire la probabilité des topics et de s'intéresser à leur distribution. Il est possible aussi de créer et de fournir l'embedding au modèle. Le modèle qui a été chargé est le *SentenceTransformer : all-MiniLM-L6-v2*.

Le travail présenté se concentre sur les unigrams. Les résultats obtenus en bi-grams n'étaient pas concluant cependant il ne s'agit guère d'une conclusion générale car le sujet est vaste et nécessite plusieurs mises au point et tests avant de tirer une telle conclusion.

Les résultats sont ceux fournis par le prétraitement *text_stpwordsEmojiPuncAtURL* qui correspond à éliminer les stopwords, les ponctuations, les emojis, les urls ainsi que les mentions d'utilisateur. Il en découle 15 clusters. La Figure 4.32 fournit les Id des clusters ainsi que la fréquence d'appartenance des documents à un de ces topics. Le Topic -1 est le plus large : il représente un ensemble où sont déposés les documents auxquels il a été difficile d'assigner un topic. Il est donc à ignorer dans un premier temps, il pourrait être utile pour filtrer dans un second temps et optimiser les résultats.
Les Figures 4.36 et 4.37 permettent de visualiser les mots-clés les plus fréquents de chaque cluster. Ces mots-clés peuvent être considérés comme étant les aspects (modulo quelques erreurs) voire parfois être confondus à un topic.
Cette première étude permet de constater la construction d'ensembles pertinents. Sans donner un nom à chaque topic, on peut pour autant interpréter le sens à partir de ses composants. Les *topic-words* du Topic 0 tournent autour de la production et de l'environnement à travers *production, steelmaking, hydrogen, emissions, carbon, etc.*. Le Topic 1 traite de l'emploi *job, business, career, experience* et d'entreprises indiennes *tatasteel, neelachal, jswsteel* en situant correctement la région *india, odisha*. Le Topic 2 traite des mesures d'antidumping : d'une part les mesures *antidumping, dgtr (directorale general of trade remedies), imports* d'autre part des

9. https://towardsdatascience.com/interactive-topic-modeling-with-bertopic-1ea55e7d73d8
10. https://towardsdatascience.com/topic-modeling-with-bert-779f7db187e6

	Topic	Count	Name
0	-1	113	-1_amp_rail_mfg_i3
1	0	80	0_hydrogen_steelmaking_emissions_carbon
2	1	52	1_india_jswsteel_orbit_odisha
3	2	34	2_subsidies_germany_steelmaking_dgtr
4	3	31	3_gupta_sapan_counsel_sapangupta
5	4	29	4_6k_technicalanalysis_chartmill_stock
6	5	22	5_construction_production_workers_cuts
7	6	20	6_arcelor_mittal_liq_stock
8	7	17	7_yara_offshore_hydrogen_wind
9	8	16	8_demand_corporation_market_nippon
10	9	14	9_openvino_intel_machine_iamintel
11	10	14	10_bullish_flow_directionally_sentiment
12	11	14	11_dunkirk_co2_lowcarbon_capture
13	12	12	12_inevitable_concerns_dollars_billions
14	13	12	13_raises_prices_coils_ton

FIGURE 4.32 – Ensemble des topics créés et leurs fréquences d'apparition.

états *china, jsw (india), germany*. Le topic 9 concerne les technologies autour de l'IA ou des entreprises : *openvino (intel distribution), iot, data, machine, intel, etc.*. Nous ne détaillons pas tous les topics, cependant nous pouvons arriver à une première conclusion : BERTopic fournit d'une façon générale de bons clusters.

A l'instar de *pyLDAvis*, un graphique interactif permet la visualisation des topics et des mots-clés en 2D. Chaque cercle indique un topic et sa taille est proportionnelle à la fréquence des topics à travers les documents (cf. Figure 4.33). Ce graphe représente aussi la distance intertopic.
Pour chaque document, il est possible de visualiser sa probabilité d'appartenance aux différents topics. Par exemple, le tweet 350 (cf. Figure 4.34) qui traite de la production d'acier et de l'environnement a plus de chance d'être classé dans l'ensemble Topic 0. Ce dernier aborde bien ces mêmes thèmes (cf. Figure 4.36). D'autres topics ont en commun les thématiques de *production* et de *CO2* d'où la distribution des probabilités de la Figure 4.35. Par ailleurs la Figure 4.38 propose une hiérarchisation des clusters.
Ce tweet est un "bon candidat". Dans le cas où le modèle a plus de difficulté à choisir correctement un topic, les probabilités sont plus faibles et assez proches. Cet indicateur permet d'avoir un critère de confiance dans les résultats proposés par le modèle.

La Figure 4.39 est une visualisation sous forme de carte thermique ou *heatmap* de la matrice de similarité des topics. Elle est basée sur la matrice de similarité cosinus entre les embeddings des topics. Pour le Topic 0, par exemple, on retrouve bien la similarité avec les Topics 7 et 11.

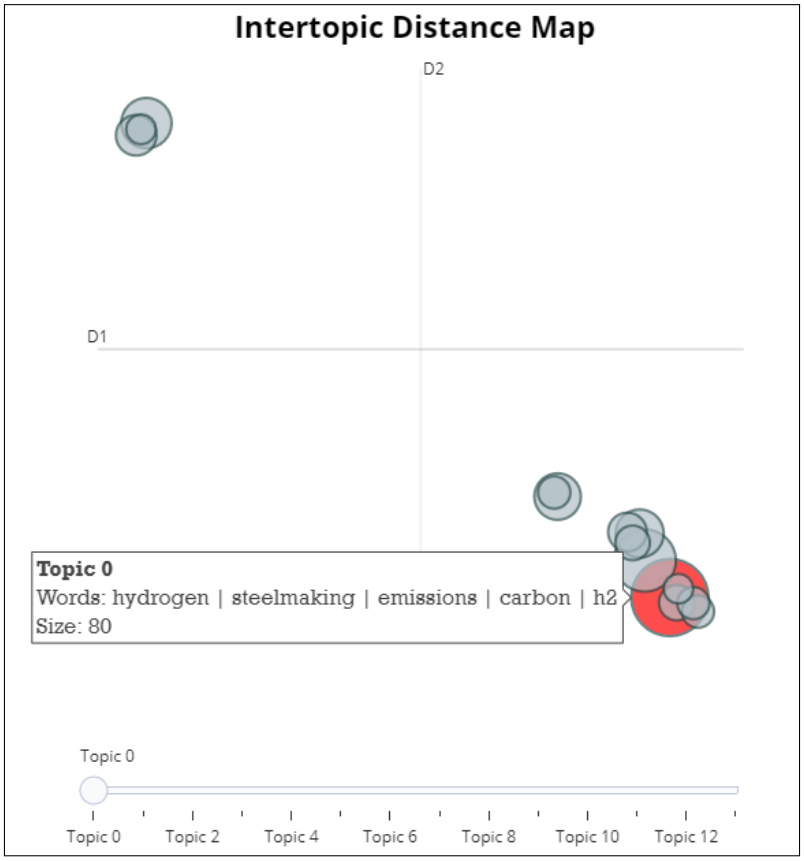


FIGURE 4.33 – Graphe des distances intertopic.

D'autres graphes intéressants peuvent être visualisés en particulier la fréquence des topics en fonction du temps. Etant donné la taille de notre corpus et son échelle temporelle, il n'y a ici aucun avantage à exploiter et afficher ces graphes.

```
model_data['full_tweet'][350]
✓ 0.4s                                                    Python
'Something new is here! #XCarb™ will bring ArcelorMittal's reduced, low and zero-carbon products,
steelmaking activities, wider initiatives and green innovation projects, into a single effort
focused on achieving #carbonneutral steelmaking. Click for more:#sourceAmel'
```

FIGURE 4.34 – Tweet 350.

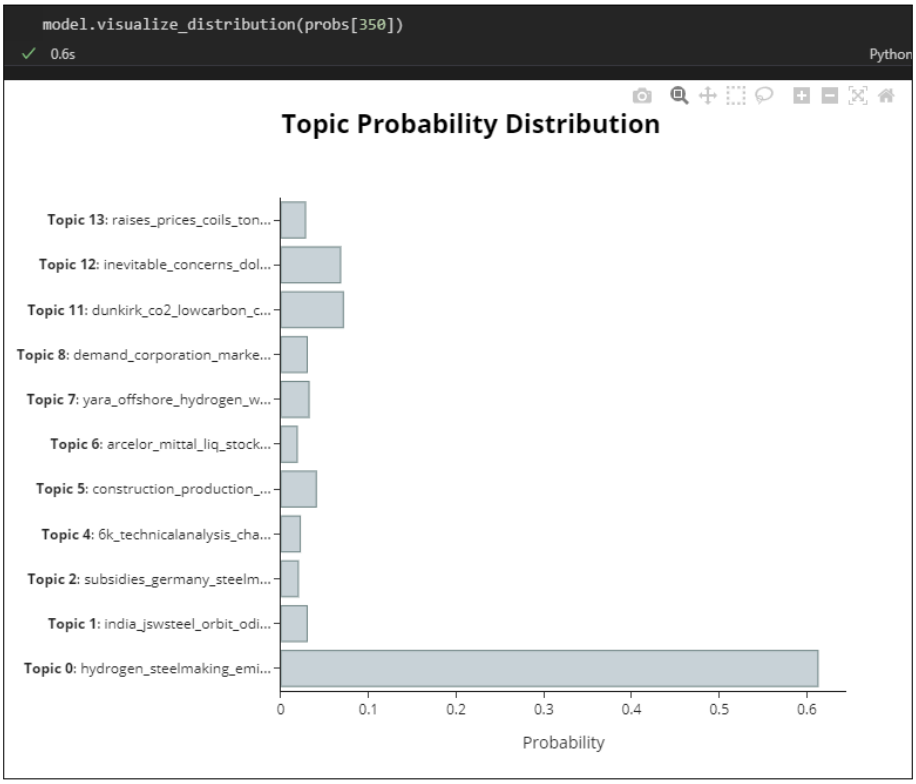


FIGURE 4.35 – Probabilité que le Tweet 350 appartienne à un des topics.

Nom de la variable VS preprocessing	@user	url	punc	emoji	stopwords	lowercase
text_emoji	OK	OK	excect (.,!?)	OK	KO	KO
textLower_emoji	OK	OK	except (.,!?)	OK	KO	OK
text_stpwordsEmoji	OK	OK	except (.,!?)	OK	OK	KO
textLower_stpwordsEmoji	OK	OK	except (.,!?)	OK	OK	OK
text_emojiPunc	OK	OK	OK	OK	KO	KO
textLower_emojiPunc	OK	OK	OK	OK	KO	OK
text _stpwordsEmoji-Punc	OK	OK	OK	OK	OK	KO
textLower _stpword-sEmojiPunc	OK	OK	OK	OK	OK	OK

TABLE 4.2 – Différents préprocessing effectués

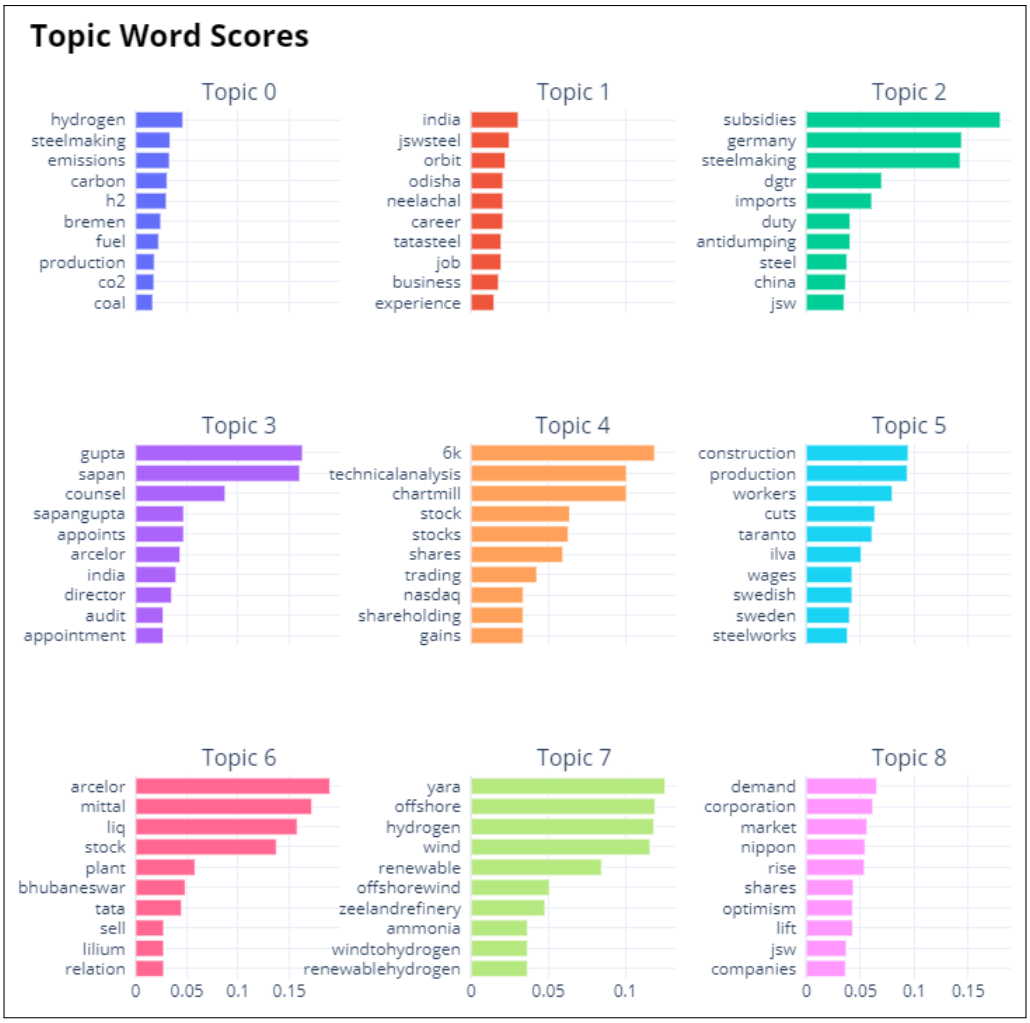


FIGURE 4.36 – Ensemble des topics créés et des mots-clés les plus représentatifs.

FIGURE 4.37 – Ensemble des topics créés et des mots-clés les plus représentatifs.

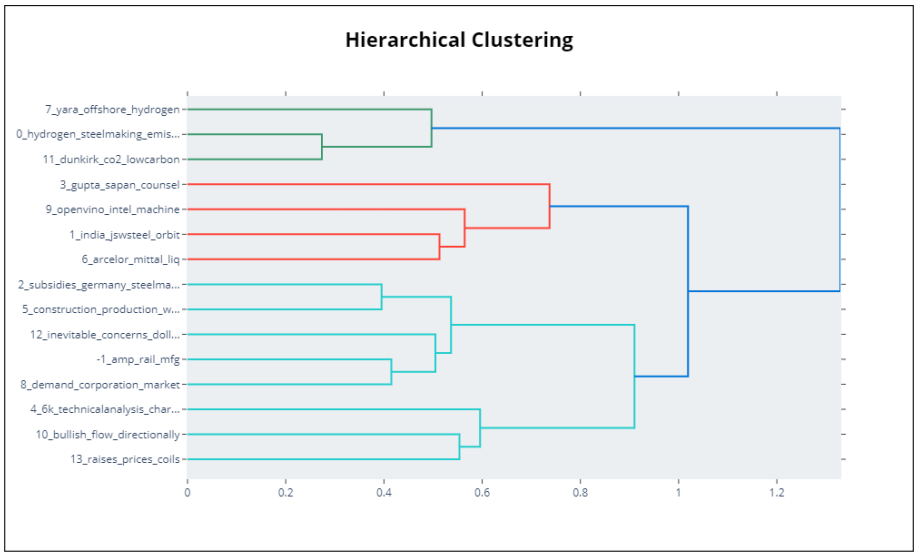


FIGURE 4.38 – Hiérarchisation des clusters.

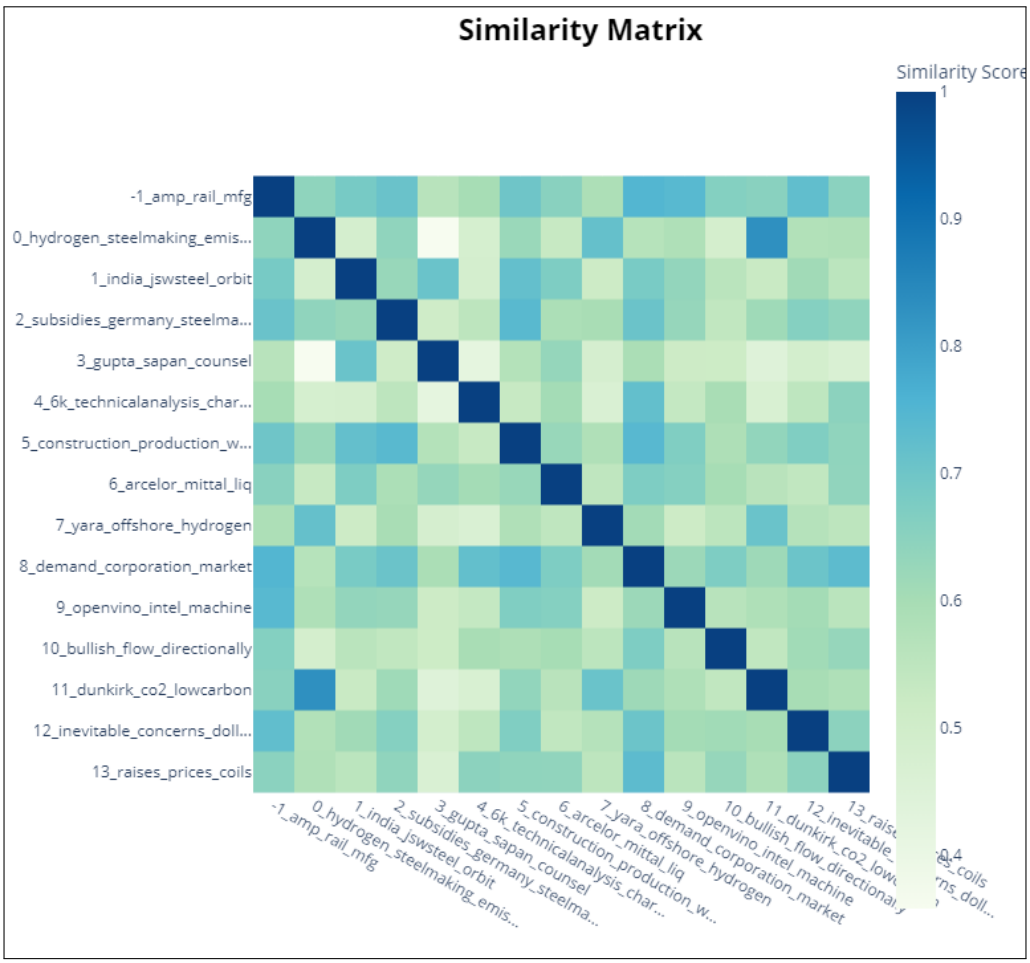


FIGURE 4.39 – Similarité entre topics.

KeyBERT

KeyBERT est une solution technique "minimale" d'extraction de mots-clés et/ou de syntagmes dont l'utilisation est simple. KeyBert exploite les embeddings de BERT et le cosinus similarité pour trouver les syntagmes d'un document les plus similaires au document lui même. Les mots les plus similaires sont identifiés comme étant ceux qui décrivent le mieux le document [Grootendorst, 2020b].

Nous ne proposons ici qu'une simple démonstration de quelques résultats obtenus suite à des tests. L'idée sous-jacente est la combinaison, l'imbrication de divers modèles dans le but d'obtenir les résultats les plus satisfaisants possibles. KeyBERT pourrait être utile pour vérifier ou filtrer les ensembles construits par BERTopic ou par la LDA.

Le modèle chargé est *all-mpnet-base-v2* [11]. Il s'agit d'un modèle *sentence-transformers* qui mappe les phrases et paragraphes dans un espace vectoriel dense de dimension 768. Ce modèle peut être utilisé pour des tâches telles que la clusterisation et la recherche sémantique. Le modèle a été entrainé sur divers datasets sur plus d'un milliard de paires.

Les Figures 4.43 & 4.44 fournissent les extractions avec et sans stopwords du tweet présenté par la Figure 4.40. Les mots-clés, surlignés, que l'on visualise dans la Figure 4.41 correspondent bien aux mots clés du tweet. Seul le mot *green* a été tronqué.

Le modèle permet également la prise en compte d'un *Vectorizer* particulier. La fonctionnalité a été testée grâce à *Scikit-learn* au travers de *CountVectorizer* et de *TfidfVectorizer*. La Figure 4.42 présente le tweet dont le résultat est fourni par la Figure 4.46 : certains mots clés sont biens extraits cependant, il manque quelques termes tels que *technology* et *machine learning*. En considérant d'autres tweets, on peut supposer que ces termes émergeraient.

En guise de première conclusion concernant *KeyBERT* : nous pouvons affirmer que les "premiers" tests s'avèrent positifs. Peu de bruits semblent être embarqués dans les solutions proposées.

```
doc
```
Python
```
"Last week, the world's second largest steel maker Arcelormittal launched a green steel carbon
offset program, and announced a plan to invest $100M a year in green steel initiatives.  "
```

FIGURE 4.40 – Exemple de tweet utilisé pour démonstration des résultats issus de *KeyBert*.

11. https://huggingface.co/sentence-transformers/all-mpnet-base-v2

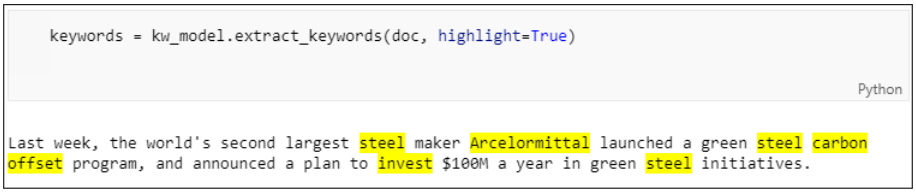


FIGURE 4.41 – Mots-clés du document surlignés.

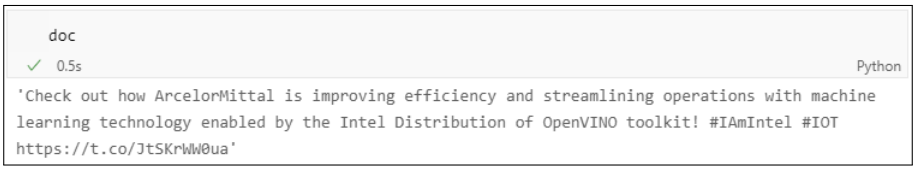


FIGURE 4.42 – Exemple de tweet utilisé pour démonstration des résultats issus de *KeyBert* et de *Vectorizer*.

```
%%time
kw_model.extract_keywords(
    doc,
    top_n=10,
    keyphrase_ngram_range=(1, 1),
    stop_words="english",
)
```

```
Wall time: 276 ms
[('arcelormittal', 0.4654),
 ('steel', 0.3599),
 ('carbon', 0.3264),
 ('offset', 0.3087),
 ('invest', 0.1598),
 ('green', 0.1515),
 ('plan', 0.1322),
 ('maker', 0.1208),
 ('initiatives', 0.1099),
 ('100m', 0.1092)]
```

FIGURE 4.43 – Extraction de mots-clés.

```
kw_model.extract_keywords(
    doc,
    keyphrase_ngram_range=(1, 1),
    stop_words=None
)
```

```
[('arcelormittal', 0.4654),
 ('steel', 0.3599),
 ('carbon', 0.3264),
 ('offset', 0.3087),
 ('invest', 0.1598)]
```

FIGURE 4.44 – Extraction de mots-clés sans stopwords.

```
vectorizer.get_feature_names()
```

```
['and',
 'arcelormittal',
 'by',
 'check',
 'distribution',
 'efficiency',
 'enabled',
 'how',
 'iamintel',
 'improving',
 'intel',
 'iot',
 'is',
 'learning',
 'machine',
 'of',
 'openvino',
 'operations',
 'out',
 'streamlining',
 'technology',
 'the',
 'toolkit',
 'with']
```

FIGURE 4.45 – Features extraits du *Vectorizer*.

```
kw_model.extract_keywords([doc], keyphrase_ngram_range=(3, 3), stop_words='english',
                          use_mmr=True, diversity=0.2,vectorizer=vectorizer)
```

Python

```
C:\ \Anaconda3\envs\py3-TF2\lib\site-packages\keybert\_model.py:130: UserWarning:
Although extracting keywords for multiple documents is faster than iterating over single
documents, it requires significantly more memory to hold all word embeddings. Use this at your
own discretion!
  warnings.warn("Although extracting keywords for multiple documents is faster "
1it [00:00, 66.68it/s]
[[('toolkit', 0.3259),
  ('openvino', 0.3724),
  ('arcelormittal', 0.4033),
  ('intel', 0.4146),
  ('iot', 0.4465)]]
```

FIGURE 4.46 – Extraction de mots-clés de KeyBert utilisant un *Vectorizer* (ti-idf, count,..).

SBERT

Sentence-BERT (SBERT) est une modification du modèle pré-entrainé BERT qui utilise des structures de réseaux siamois. Les plongements de phrases sémantiques sont comparés à l'aide du cosinus similarité. Ce modèle réduit l'effort pour trouver la paire de phrases la plus similaire de 65 heures avec BERT ou RoBERTa à environ 5 secondes tout en maintenant la précision de BERT [Reimers and Gurevych, 2019].

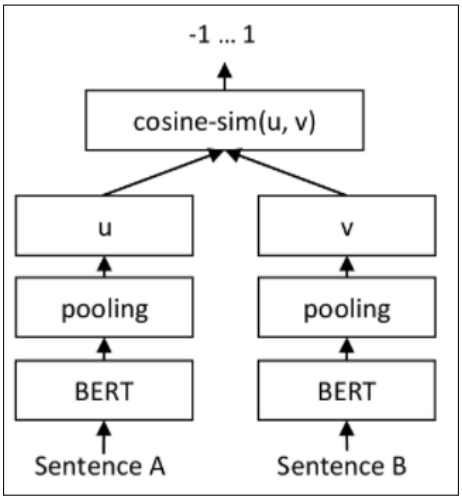


FIGURE 4.47 – Architecture de SBERT.

Nous ne proposons ici qu'une simple démonstration d'un résultat obtenu. L'idée sous-jacente est la combinaison, l'imbrication de divers modèles dans le but d'obtenir les résultats les plus satisfaisants possibles. SBERT pourrait être utile pour vérifier ou filtrer les ensembles construits à travers un seuil qui serait imposé sur le score de similarité.

Une des listes les plus exhaustives de modèles pré-entrainées est fournie par le lien suivant [12]. Un menu permet de filtrer en fonction des tâches (NLP : résumé, Q-A, Classification,... , Audio : Text-to-Speech, Voice Activity Detection,.., Computer vision : Objet Detection, Text-to-Image...), des librairies (PyTorch, TensorFlow, JAX, Spacy, Stanza, Keras,..), des Datasets (wikipedia, glue, bookcorpus,..), des langues (plus de 164 langues) , etc. Il suffit alors de sélectionner des modèles, de les charger à travers des *SentenceTransformer* et de les tester sur le sujet en question.

Pour comparer les résultats, nous utilisons les plongements générés par *SBERT* et le cosinus similarité de *Scikit-Learn* : la Figure 4.48 montre le résultat obtenu pour le cas du premier tweet comparé au reste des tweets du corpus. Nous appliquons alors un seuil de 0.5 afin de ne conserver que les tweets dont le score est supérieur (cf. Figure 4.49). Quelques tweets ont été sélectionnés pour vérifier les résultats : cela semble cohérent et bon. Nous gardons alors en tête que SBERT pourrait aider dans la construction de clusters optimisés.

D'autres modèles existent notamment *BERTweet* : il serait intéressant de continuer à creuser ces pistes, à prendre en main ces modèles et à les comparer pour ensuite

12. https://huggingface.co/models?library=sentence-transformers

```
cos_sim = cosine_similarity(
    [sentence_embeddings[0]],
    sentence_embeddings[1:]
).tolist()
```

Python

```
df_sim = pd.DataFrame(zip(docs[1:],cos_sim[0]),columns=['tweet','similarity'])
df_sim
```

Python

	tweet	similarity
0	@AssindMantova 50/50 joint venture Arce...	0.485852
1	ArcelorMittal Investing Nearly $400 Millio...	0.478073
2	ArcelorMittal World Green Steel Vouche...	1.000000
3	$MT ArcelorMittal volume normal directi...	0.317941
4	$MT ArcelorMittal volume normal directi...	0.285298
...	...	...
474	case missed ArcelorMittal low-carbon s...	0.716278
475	September 2019 Lex Greensill confidant ...	0.449786
476	@TodaDogs @steelanol @LanzaTech @inea_eu @...	0.533882
477	@ArcelorMittal Sir belong poor family ...	0.393147
478	@ArcelorMittal committed development rol...	0.790232

479 rows × 2 columns

FIGURE 4.48 – Comparaison en terme de cosinus similarité du premier tweet aux autres tweets.

```
df_cos = df_sim.loc[df_sim['similarity'] >= 0.5]
df_cos.sort_values(by='similarity', ascending=False)
```

	tweet	similarity
2	ArcelorMittal World Green Steel Vouche...	1.000000
349	new #XCarb bring ArcelorMittal reduc...	0.874639
411	#XCarb unites @ArcelorMittal reduced ...	0.867296
234	ArcelorMittal Advancing Green Steel Produ...	0.865083
451	ArcelorMittal creates new green steel ini...	0.861397
...	...	...
359	considering new technology help mitigat...	0.502805
409	considering new technology help mitigat...	0.502805
408	considering new technology help mitigat...	0.502805
403	considering new technology help mitigat...	0.502805
436	ArcelorMittal taken bulk Uttam Galva d...	0.500337

223 rows × 2 columns

FIGURE 4.49 – Filtrage des tweets sur le cosinus similarité via un seuil de 0.5.

sélectionner celui qui répond le mieux à notre sujet.

.

4.5 Analyse d'opinions à l'échelle des mots et des syntagmes

Pour rappel, nous nous sommes intéressés jusqu'à présent d'une part à l'analyse du sentiment à l'échelle du document, et d'autre part à l'extraction d'aspects et de topics.

Une analyse plus fine du sentiment à l'échelle des mots et/ou de syntagmes est également possible. Cette tâche fait partie des différents traitements qui peuvent être effectués en *ABSA* : il s'agit d'associer une orientation du sentiment (positif, négatif ou neutre) à chaque aspect identifié. Il n'existe pas de méthodes à proprement parler qui répondent directement au besoin. Cette thématique de recherche est d'actualité en *NLP*.

Cette section, étant donné le temps imparti, se réduit à quelques premiers tests et à quelques idées et directions émises.

4.5.1 Analyse syntaxique et spaCy

Nous cherchons à mettre en place dans cette étude les différentes briques qui permettront d'arriver à une analyse la plus fine possible. Analyse qui permettra d'extraire un maximum d'informations sur des thématiques données à travers ses objets et l'orientation d'opinion sur ces derniers. Les sections précédentes permettent d'arriver à l'extraction d'aspects et à un étiquetage globale du sentiment à l'échelle du document.

Il est proposé ici d'extraire le sentiment associé à un aspect. L'idée première a été d'utiliser les arbres de dépendances et certaines catégories morphosyntaxiques. Nous avons donc émis comme première hypothèse que les aspects seraient des tokens de type *NOUN* voire de type *PRON*.
N.B. : Les pronoms peuvent être considérés dans un second temps. Souvent ils sont identifiés à It, I, you, myself, etc. ce qui est difficile à interpréter.

Nous avons utilisé le POS Tagging pour identifier notamment les noms, les verbes, les modifieurs nominaux et verbaux (adjectifs et adverbes). Plusieurs combinaisons et tests ont été éprouvés utilisant les dépendances syntaxiques, parcourant les enfants des syntagmes nominaux etc. Cependant, assez rapidement nous avons été confrontés au cumul de problèmes :

— sensibilité à la casse de spaCy (cf. les Figures 4.50 et 4.51),
— analyse syntaxique erronée,
— dépendance syntaxique erronée,
— difficulté à décomposer un document en plusieurs phrases spécialement quand les ponctuations sont manquantes,
— structure même des tweets insérant des emojis, des mentions d'utilisateurs, des urls, des hashtags parfois eux mêmes composés (cf. Fig. 4.51c et 4.51c),
— fautes typographiques etc.

En sortie les résultats portaient trop de bruits. La piste spaCy sous cet angle parait peu satisfaisante.

```
print(f"Sentence: {text_pre.lower()} \n")
doc = nlp(text_pre.lower())
for chunk in doc.noun_chunks:
    print(chunk.text," | ", chunk.root.pos_," | ", chunk.root.text," | ", chunk.root.dep_," | ",
        chunk.root.head.text," | ", chunk.root.head.pos_)
```

✓ 0.6s Python

```
Sentence: arcelormittal has world's first 'green' steel voucher program

world's first 'green' steel voucher program  |  NOUN  |  program  |  dobj  |  has  |  VERB
```

(a) Syntagme nominal sur un tweet en lowercase

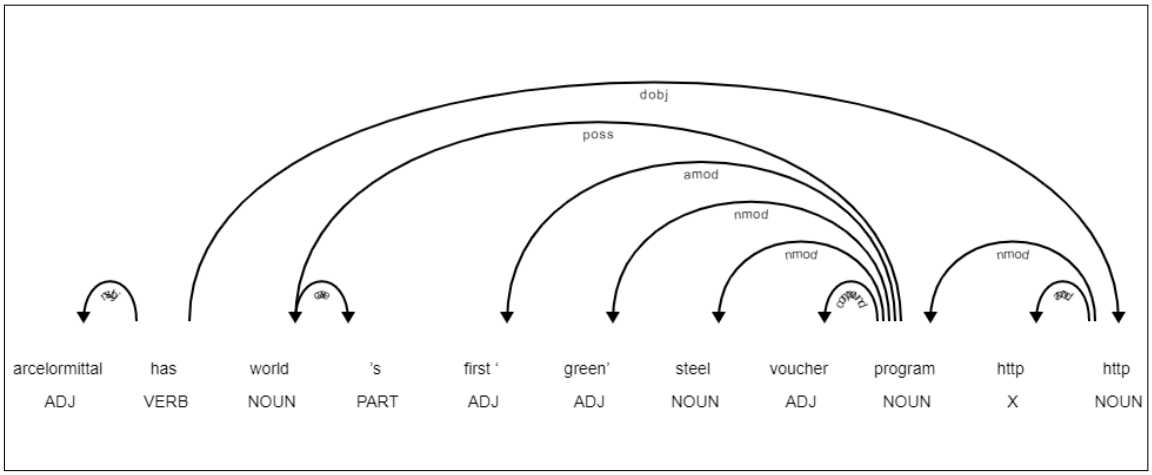
```
print(f"Sentence: {text_pre} \n")
doc = nlp(text_pre)
for chunk in doc.noun_chunks:
    print(chunk.text," | ", chunk.root.pos_," | ", chunk.root.text," | ", chunk.root.dep_," | ",
        chunk.root.head.text," | ", chunk.root.head.pos_)
```

✓ 0.4s Python

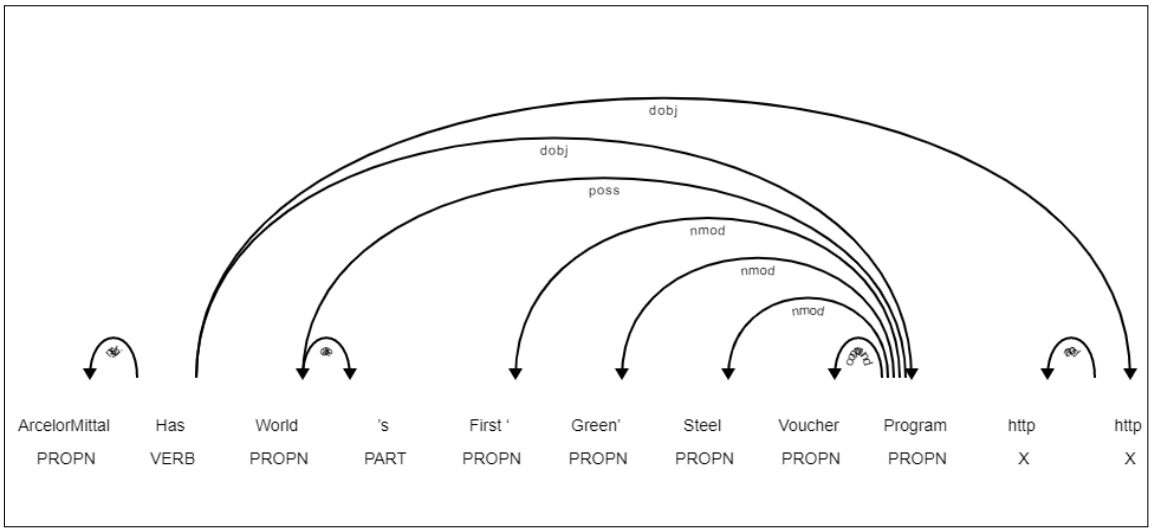
```
Sentence: ArcelorMittal Has World's First 'Green' Steel Voucher Program

ArcelorMittal  |  PROPN  |  ArcelorMittal  |  nsubj  |  Has  |  VERB
World's First 'Green' Steel Voucher Program  |  PROPN  |  Program  |  dobj  |  Has  |  VERB
```

(b) Syntagme nominal sur un tweet

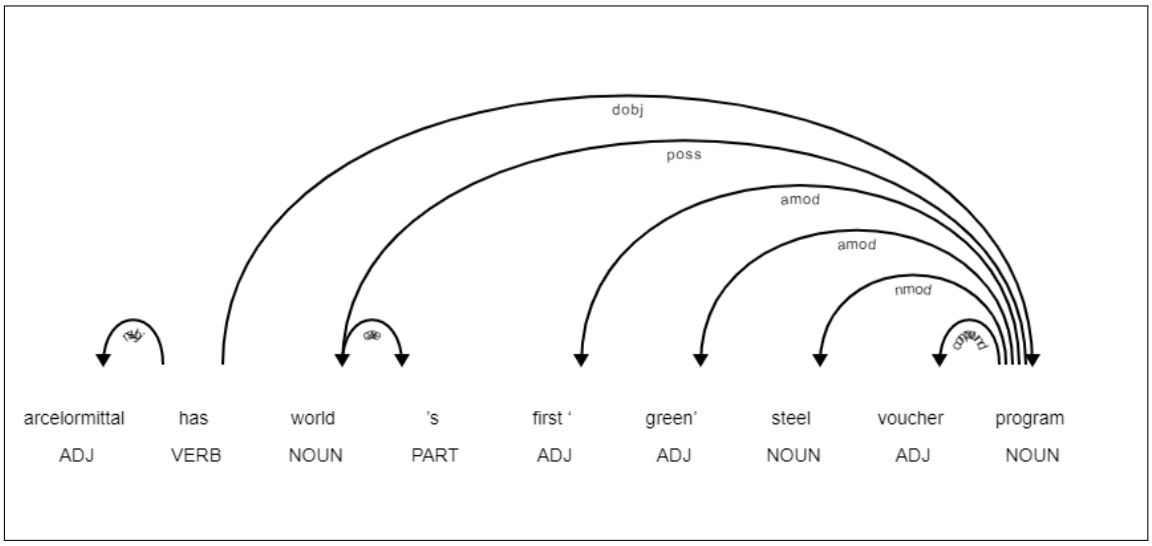
FIGURE 4.50 – Extraction des noun_chunks avec spaCy 3.0.5



(a) Tweet en minuscule



(b) Tweet quasi-brut



(c) Tweet sans URLs

FIGURE 4.51 – Arbres de dépendance avec/sans prise en compte de la casse

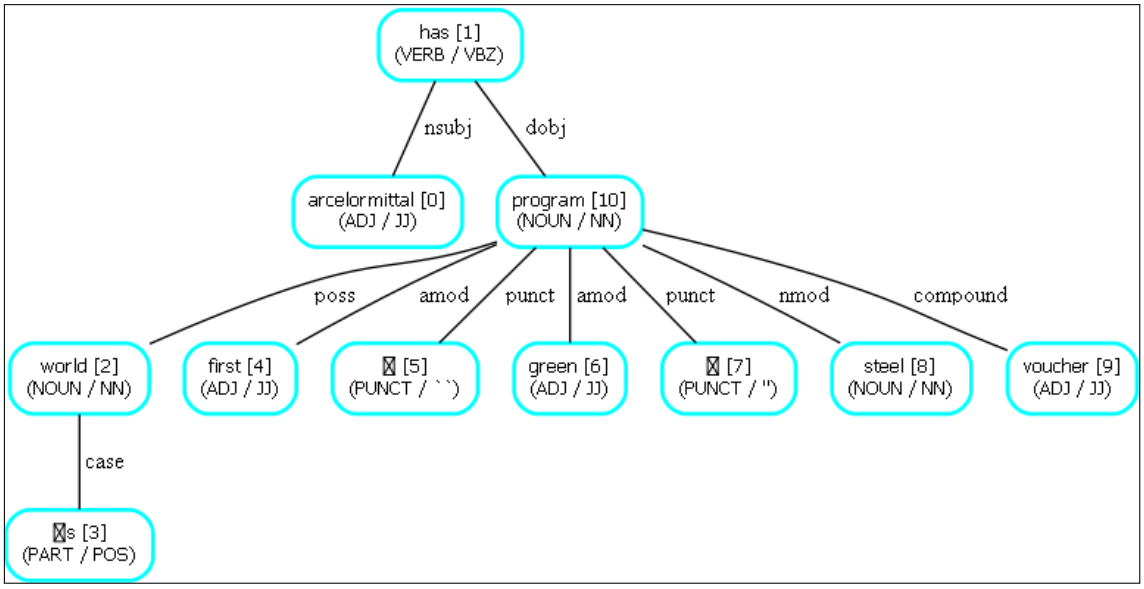


FIGURE 4.52 – Arbre de dépendance.

4.5.2 Lexicon-based

Cette piste n'a pas été exploitée mais les idées sont posées. Il faudrait :

— utiliser les résultats des modèles précédents à savoir les topics et les mots-clés pour identifier les aspects dans les documents.
  — Lors de cette étape, il serait judicieux d'utiliser un schéma d'encodage des annotations de type BIO2. L'article [Gandhi and Attar, 2020] propose une méthode basée sur les modèles CRF et Bi-LSTM pour extraire les aspect-terms. Le schéma intégré est BIO, cependant comme nous avons pu voir en fouilles de texte : le schéma BIO2 est le plus employé par les CRF nous proposons de basculer sur ce type d'encodage.
— créer des lexiques spécifiques par thématiques :
  — un vocabulaire de l'emploi (s'inspirer d'*Indeed*® pour les catégories, etc.),
  — un vocabulaire de l'environnement et de l'écologie (s'inspirer de la campagne *DEFT 2015* consacrée à l'analyse automatique en sentiments, opinions, émotions de twitts (notamnt dans le domaine de l'écologie) dont le corpus de développement a été réannoter par l'*ERTIM*.),
  — un vocabulaire des nouvelles technologies, etc.
  — extraire et compléter le glossaire ArcelorMittal (à organiser par concept à l'instar des Entités Nommées afin de repérer les personnes, les lieux, les organisations,...). La reconnaissance d'entités nommées (NER) pourrait alors être appliquée et faciliter l'annotation.
— utiliser des lexiques d'opinions négatifs et positifs : celui de Minqing Hu et Bing Liu est le plus répendu. Il contient volontairement des mots dont l'écriture est erroné : ils correspondent à ceux qui apparaissent fréquemment dans les contenus des réseaux sociaux (cf. *negative-words.txt* et *positive-words.txt*). Les permissions d'utilisation sont accordées quelque soit le cadre. Dans la mesure où nous traitons des documents *tweets*, le lexique *SemEval-2015 English Twitter Lexicon* devrait compléter le lexique précédent. En plus des mots, leur sentiment positif/négatif associé est donné. Il contient par ailleurs des *#hashtags*. Le lexique peut-être utilisé librement pour des sujets de recherche pu-

rement académique. Il est facilement disponible sur le net. Pour s'assurer des droits d'utilisation dans un cadre plus étendu : il suffit de contacter le Dr. M. Saif du *National Research Council Canada (NRC)*.

Cette liste peut être complétée mais pour une première approche les points les plus importants sont amplement pris en compte.

4.6 Conclusion

D'ores et déjà, certaines pistes sont à exploiter et d'autres à abandonner.
La première partie de ce chapitre se concentrait sur l'analyse de sentiments à l'échelle du document. Nous nous sommes intéressés dans un premier temps à des approches de type lexicon-based. Deux modèles, que l'on retrouve fréquemment dans la littérature, ont été utilisés : Textblob et VADER. Une étude comparative de ces modèles à révéler qu'ils évaluaient un tweet sur deux différemment. Il s'avère que ces modèles rencontrent quelques difficultés face au vocabulaire de nos tweets. Nous nous sommes ensuite dirigés vers des méthodes pré-entrainés de type BERT : certains résultats semblent intéressants cependant les modèles ne s'accordent pas sur tous les résultats. La piste reste à être creusée mais il semblerait qu'il soit nécessaire que l'humain vérifie, valide ou rejette certains résultats à l'instar de l'annotation.
La seconde partie de ce chapitre se concentrait sur l'analyse de sentiments à l'échelle des entités et des aspects. Des techniques de topic modeling ont été appliqués, BERTopic semble être un bon candidat : des clusters cohérents sont créés. Ces derniers pourraient nous permettre l'extraction d'aspects. Le socle est posé. Le dernier point abordé s'attache à lier à l'aspect une orientation de polarité. Nous avons eu recours à l'analyse syntaxique cependant la nature même des tweets a fait échouer l'analyseur syntaxique.

CHAPITRE 5

DISCUSSION & CONCLUSION

Sommaire

5.1 Discussion & Conclusion

L'analyse de sentiments est un domaine très large qui peut aller d'un simple problème de classification supervisé dont le corpus est annoté à un problème appartenant à des thématiques de recherches d'actualité tel est le cas des sujets s'intéressant à extraire des aspects tout en leur associant une orientation de polarité.

L'analyse de sentiments à l'échelle du document dans le cas d'un corpus non annoté n'a à ce stade de l'étude pas encore trouvé une solution qui exclut l'intervention humaine du moins dans notre cas. En effet les résultats de méthodes de type lexicon-based n'ont pas été convaincants. Nous attendons de BERT une meilleure réponse mais nous avons d'ores et déjà relevé des étiquettes erronées : quel est le seuil d'erreurs acceptable et comment l'évalue-t-on ? Il semble que l'on ne puisse se passer de l'intervention humaine et de sa validation voire de son annotation aussi restreinte soit-elle.

L'analyse de sentiments à l'échelle des aspects traite deux grands sujets : l'extraction d'informations que sont les aspects voire les entités également et la capacité à associer à ce même aspect un sentiment de polarité qui lui même est porté par un autre élément du texte.
Concernant l'extraction d'informations, nos pistes de topic modeling sauront sans aucun doute répondre au sujet. Que se soit à travers une approche LDA ou une approche BERTopic. Pour l'approche LDA l'article [Boumedyen Billami et al., 2020] conclue qu'au moyen d'un algorithme k-Means les résultats de classifications sont de qualité comparable avec celle effectuée manuellement par un expert (combinaison LDA/k-Mean).
Concernant l'association aspect/polarité, nous n'avons pu exploiter que la piste de l'analyse syntaxique. Cette piste a été mise en échec : la structure même des tweets a généré trop de bruits et d'erreurs ne permettant pas d'arriver à travers les arbres de dépendance aux bons éléments. Des pistes ont été proposées dans la Section 4.5.2. Une première pourrait être d'établir un lexique d'opinions et de considérer une fenêtre entre l'aspect et l'opinion.

BIBLIOGRAPHIE

[Abdi et al., 2020] Abdi, A., Idris, N., Maitama, J., Shuib, L., and Fauzi, R. (2020). A systematic review on implicit and explicit aspect extraction in sentiment analysis. *IEEE Access*, 8:194166–194191. – Cité page 24.

[Abdullah et al., 2017] Abdullah, N. A. S., Shaari, N., and Rahman, A. (2017). Review on sentiment analysis approaches for social media data. *Journal of Engineering and Applied Sciences*, 12:462–467. – Cité pages 4, 20 et 21.

[Amara et al., 2020] Amara, A., Hadj Taieb, M. A., and Ben Aouicha, M. (2020). Multilingual topic modelling for tracking covid-19 trends based on facebook data analysis. – Cité page 23.

[Barbieri et al., 2020] Barbieri, F., Camacho-Collados, J., Espinosa Anke, L., and Neves, L. (2020). TweetEval: Unified benchmark and comparative evaluation for tweet classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1644–1650, Online. Association for Computational Linguistics. – Cité page 54.

[Bonta et al., 2019] Bonta, V., Kumaresh, N., and Janardhan, N. (2019). A comprehensive study on lexicon based approaches for sentiment analysis. *Asian Journal of Computer Science and Technology*, 8:1–6. – Cité pages 21 et 22.

[BOULLIER and LOHARD, 2012] BOULLIER, D. and LOHARD, A. (2012). *Opinion mining et ?Sentiment analysis : Méthodes et outils*. – Cité page 13.

[Boumedyen Billami et al., 2020] Boumedyen Billami, M., Bortolaso, C., and Derras, M. (2020). Extraction de thèmes d'un corpus de demandes de support pour un logiciel de relation citoyen. In Benzitoun, C., Braud, C., Huber, L., Langlois, D., Ouni, S., Pogodalla, S., and Schneider, S., editors, *6e conférence conjointe Journées d'Études sur la Parole (JEP, 33e édition), Traitement Automatique des Langues Naturelles (TALN, 27e édition), Rencontre des Étudiants Chercheurs en Informatique pour le Traitement Automatique des Langues (RÉCITAL, 22e édition). Volume 2 : Traitement Automatique des Langues Naturelles*, pages 155–163, Nancy, France. ATALA. – Cité page 81.

[Colón-Ruiz and Segura-Bedmar, 2020] Colón-Ruiz, C. and Segura-Bedmar, I. (2020). Comparing deep learning architectures for sentiment analysis on drug reviews. *Journal of Biomedical Informatics*, 110:103539. – Cité page 25.

[D'Andrea et al., 2015] D'Andrea, A., Ferri, F., Grifoni, P., and Guzzo, T. (2015). Approaches, tools and applications for sentiment analysis implementation. *International Journal of Computer Applications*, 125:26–33. – Cité page 20.

[Darwich et al., 2019] Darwich, M., Mohd Noah, S. A., Omar, N., and Osman, N. (2019). Corpus-based techniques for sentiment lexicon generation: A review. *Journal of Digital Information Management*, 17:296. – Cité page 19.

[Gandhi and Attar, 2020] Gandhi, H. and Attar, V. (2020). Extracting aspect terms using crf and bi-lstm models. *Procedia Computer Science*, 167:2486–2495. International Conference on Computational Intelligence and Data Science. – Cité pages 4, 24, 25 et 78.

[Grootendorst, 2020a] Grootendorst, M. (2020a). Bertopic: Leveraging bert and c-tf-idf to create easily interpretable topics. – Cité page 63.

[Grootendorst, 2020b] Grootendorst, M. (2020b). Keybert: Minimal keyword extraction with bert. – Cité page 71.

[Khairiyah Mohamed and Carol Anne, 2021] Khairiyah Mohamed, R. and Carol Anne, H. (2021). Leveraging twitter data to understand public sentiment for the covid?19 outbreak in singapore. *International Journal of Information Management Data Insights*, 1(2). – Cité page 23.

[Khan et al., 2020] Khan, R., Shrivastava, P., Kapoor, A., Tiwari, A., and Mittal, A. (2020). Social media analysis with ai: Sentiment analysistechniques for the analysis of twitter covid-19 data. – Cité page 21.

[Lagrari and Elkettani, 2021] Lagrari, F.-e. and Elkettani, Y. (2021). Traditional and Deep Learning Approaches for Sentiment Analysis: A Survey. *Advances in Science, Technology and Engineering Systems Journal*, 6(5):1–7. – Cité pages 25, 27 et 31.

[Liu and Zhang, 2013] Liu, B. and Zhang, L. (2013). A survey of opinion mining and sentiment analysis. *Mining Text Data*, pages 415–463. – Cité pages 17 et 19.

[László and Attila, 2021] László, N. and Attila, K. (2021). Social media sentiment analysis based on covid-19. *Journal of Information and Telecommunication*, 5(1):1 – 15. – Cité pages 22 et 49.

[Magué et al., 2020] Magué, J.-P., Rossi-Gensane, N., and Halté, P. (2020). De la segmentation dans les tweets : signes de ponctuation, connecteurs, émoticônes et émojis. *Corpus*. – Cité page 20.

[Mrityunjay et al., 2021] Mrityunjay, S., Jakhar, A., and Pandey, S. (2021). Sentiment analysis on the impact of coronavirus in social life using the bert model. *Social Network Analysis and Mining*, 11. – Cité page 25.

[Patil and Pratibha, 2016] Patil, P. and Pratibha, Y. (2016). Sentiment analysis levels and techniques: A survey. *International Journal of Innovations in Engineering and Technology*, 6:123–128. – Cité pages 18 et 19.

[Ravi, 2015] Ravi, K. (2015). A survey on opinion mining and sentiment analysis: Tasks, approaches and applications. *Knowledge-Based Systems*, 89:14–46. – Cité page 19.

[Reimers and Gurevych, 2019] Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. – Cité page 74.

[Valette, 2016] Valette, M. (2016). Analyse statistique des données textuelles et traitement automatique des langues. Une étude comparée. In Mayaffre, D., Poudat, C., Vanni, L., Magri, V., and Follette, P., editors, *International Conference on Statistical Analysis of Textual Data (JADT2016)*, volume 2 of *Proceedings of 13th International Conference on Statistical Analysis of Textual Data, 7-10 June 2016, Nice (France)*, pages 697–706, Nice, France. – Cité page 30.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc. – Cité page 25.

ANNEXE A

DOCUMENTATION

A.1 wordnet

Lorsque l'on cherche un mot faisant partie du lexique de WordNet, on utilise la fonction *s*ynset(). Le mot est alors retourné sous une forme composée de 3 parties : *s*ynset = Lemma.POS.NN où *N*N est une clé de sens étant donné qu'un mot peut avoir plusieurs significations ou définitions.

Voici quelques exemples applicables à notre thématique :

Dans une démarche de recherche d'identification d'aspects voire d'opinions, une recherche des hypernymes, hyponymes, co-hyponymes, etc... pourrait aider ?

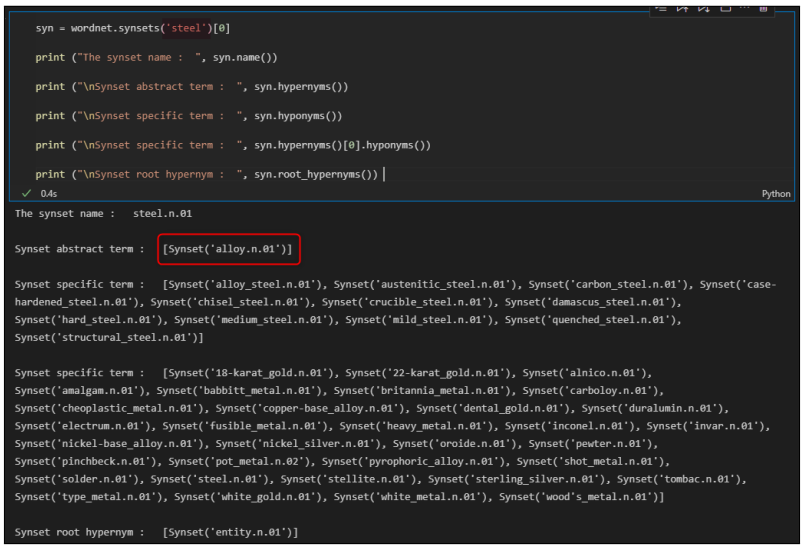


FIGURE A.1 – Résultats de recherche dans le lexique WordNet du term *s*teel.

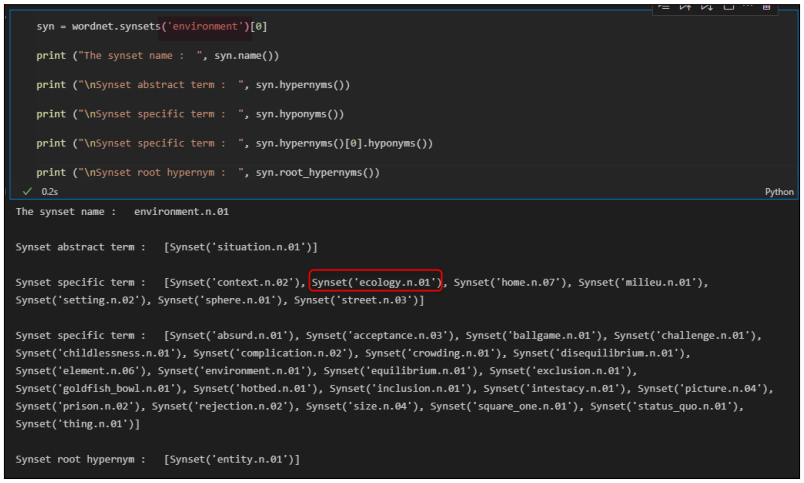


FIGURE A.2 – Résultats de recherche dans le lexique WordNet du term *e*nvironment.

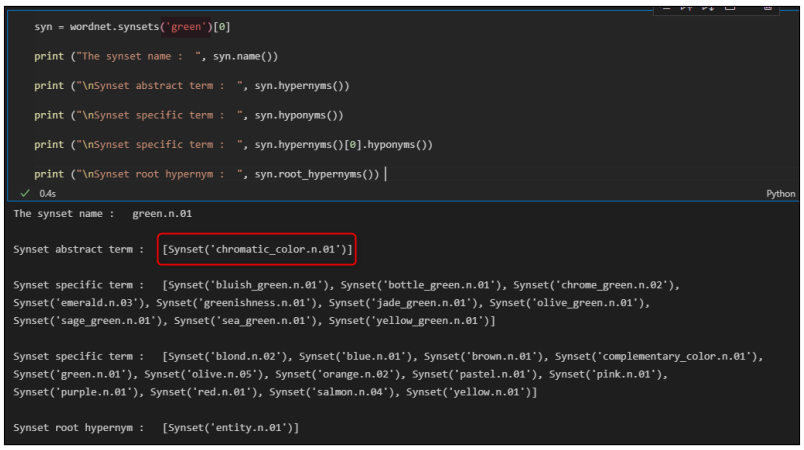


FIGURE A.3 – Résultats de recherche dans le lexique WordNet du term *g*reen.

A.2 Formules mathématiques

Le rappel (formule A.1) mesure le nombre d'éléments correctement étiquetés par le système (vrais positifs) rapporté au nombre d'éléments étiquetés dans la référence (vrais positifs et faux négatifs).

$$\text{Rappel} = \frac{\text{vrais positifs}}{\text{vrais positifs + faux négatifs}} \tag{A.1}$$

La précision (formule A.2) mesure le nombre d'éléments correctement étiquetés par le système (vrais positifs) rapporté au nombre total d'éléments étiquetés par le système (vrais et faux positifs).

$$\text{Précision} = \frac{\text{vrais positifs}}{\text{vrais positifs + faux positifs}} \tag{A.2}$$

La F-mesure (formule A.3) est la moyenne harmonique pondérée du rappel et de la précision. La valeur accordée à β permet de pondérer le rappel ou la précision, ou d'équilibrer les deux mesures (avec $\beta = 1$).

$$\text{F-mesure} = \frac{(1+\beta^2) \times \text{ précision } \times \text{ rappel}}{\beta^2 \times \text{ précision } + \text{ rappel}} \tag{A.3}$$

A.3 Paramètres modèle BERT

La figure A.4 présente les paramètres modèles utilisés. Ils peuvent être optimisés pour améliorer les clusters. La Figure A.5 fournit le prototype du modèle *KeyBert* et

Parameters of a Topic Mode

```
model.get_params()
```

✓ 0.5s Python

```
{'calculate_probabilities': True,
 'embedding_model': <bertopic.backend._sentencetransformers.SentenceTransformerBackend at 0x29745e8c248>,
 'hdbscan_model': HDBSCAN(min_cluster_size=10, prediction_data=True),
 'language': 'english',
 'low_memory': False,
 'min_topic_size': 10,
 'n_gram_range': (1, 1),
 'nr_topics': None,
 'seed_topic_list': None,
 'top_n_words': 10,
 'umap_model': UMAP(angular_rp_forest=True, dens_frac=0.0, dens_lambda=0.0, low_memory=False,
     metric='cosine', min_dist=0.0, n_components=5),
 'vectorizer_model': CountVectorizer(),
 'verbose': False}
```

FIGURE A.4 – Paramètres du modèle.

donc ces paramètres.

```
kw_model.extract_keywords?

Signature:
kw_model.extract_keywords(
    docs: Union[str, List[str]],
    candidates: List[str] = None,
    keyphrase_ngram_range: Tuple[int, int] = (1, 1),
    stop_words: Union[str, List[str]] = 'english',
    top_n: int = 5,
    min_df: int = 1,
    use_maxsum: bool = False,
    use_mmr: bool = False,
    diversity: float = 0.5,
    nr_candidates: int = 20,
    vectorizer: sklearn.feature_extraction.text.CountVectorizer = None,
    highlight: bool = False,
    seed_keywords: List[str] = None,
) -> Union[List[Tuple[str, float]], List[List[Tuple[str, float]]]]
Docstring:
Extract keywords/keyphrases

NOTE:
    I would advise you to iterate over single documents as they
    will need the least amount of memory. Even though this is slower,
    you are not likely to run into memory errors.

Multiple Documents:

show more (open the raw output data in a text editor) ...

Returns:
    keywords: the top n keywords for a document with their respective distances
              to the input document
```

FIGURE A.5 – Paramètres du modèle KeyBert.

A.4 Analyse du corpus

	Token	OccurenceNb
0	.	541
1	\n	404
2	,	386
3	the	308
4	to	262
5	ArcelorMittal	261
6	of	246
7	and	221
8	in	201
9	for	148

FIGURE A.6 – Les 10 premiers tokens.

	Currency code	Currency symbol
0	USD	$
1	AUD	$
2	EUR	€
3	CAD	$
4	GBP	£
5	HKD	HK$
6	INR	₨
7	NGN	₦
8	ZWD	Z$
9	JPY	¥

FIGURE A.7 – La liste des monnaies retenues pour dénombrer les tokens.

	emoji	emoji_text
0	®	registered
1	FR	flag: France
2	👉	backhand index pointing right
3	🔒	locked
4	🤔	thinking face
5	💰	money bag
6	📰	newspaper
7	👇	backhand index pointing down
8	👍	thumbs up
9	™	trade mark
10	🐾	paw prints
11	💜	purple heart
12	🚀	rocket
13	ZA	flag: South Africa
14	🤓	nerd face
15	💉	syringe
16	💪	flexed biceps
17	🙄	face with rolling eyes
18	👏	clapping hands
19	🚧	construction
20	⬇	down arrow
21	⤵	right arrow curving down
22	🌍	globe showing Europe-Africa
23	👍🏻	thumbs up: light skin tone
24	👌🏻	OK hand: light skin tone

FIGURE A.8 – Emojis présents dans le corpus.

INDEX