Dauphine | PSL
UNIVERSITÉ PARIS

afia
C F A
#RévélateurDeTalents

# Mémoire

Année universitaire 2020 / 2021

---

# La synthèse d'opinions avec l'analyse des sentiments par aspect

---


Université Paris-Dauphine

**Etudiant :** Yves Tran

**Tuteur enseignant :** Alexandre Allauzen

**Maitre d'apprentissage :** Quentin Desrousseaux

**Entreprise d'accueil :** Ministère de l'Économie, des Finances et de la Relance

**Formation :** Intelligence Artificielle, Systèmes, Données


6 Septembre 2021

# Mention

L'Université de Dauphine n'entend donner aucune approbation ni improbation aux opinions émises dans ce mémoire qui doivent être considérées comme propres à leur auteur.

# Remerciements

Je tiens tout d'abord à remercier mon maître d'apprentissage Monsieur Quentin Desrousseaux, Lead Data Scientist au sein du Bercy-Hub d'avoir accepté de m'encadrer dans cette étude. Ses encouragements, sa confiance et son soutien m'ont permis d'avancer tout au long de l'année.

Je remercie les membres de l'équipe du Bercy-Hub de m'avoir offert l'opportunité de prendre part à leurs projets.

D'autre part, je souhaite adresser mes remerciements à mon tuteur enseignant Monsieur Alexandre Allauzen, professeur à l'ESPCI ainsi qu'à l'Université Paris-Dauphine et chercheur au LAMSADE, pour ses conseils et son accompagnement durant mon année d'alternance.

Cette année d'étude m'a permis de progresser dans les domaines de l'Intelligence Artificielle et de la Science des données et je remercie vivement mes professeurs pour leur pédagogie ainsi que pour la qualité de leur enseignement.

Enfin, j'adresse mes derniers remerciements à mes proches pour leur soutien sans qui je n'aurais pu arriver à bout de ce travail.

# Résumé

Dans le cadre de ce Master en Intelligence Artificielle, Systèmes, Données au sein de l'Université Paris-Dauphine, un travail de mémoire doit être effectué visant à mener un projet mêlant Intelligence Artificielle et Science des données. Le sujet retenu s'inscrit dans le contexte du traitement automatique du langage, plus précisément sur l'analyse des sentiments par aspect. Il a pour but d'employer les méthodes d'apprentissage automatique afin de concevoir un service pour synthétiser l'opinion exprimée au sein des différents aspects abordés. Cela permet un gain de temps considérable dès lors que les données à analyser sont trop volumineuses pour être traitées manuellement.

Ce document propose une approche permettant d'affiner l'analyse des sentiments au sein d'un corpus pour obtenir la répartition des opinions à travers différents aspects d'un domaine. Elle se décompose en deux étapes dont la première est d'extraire des fragments de texte isolant le sentiment. La deuxième étape consiste à réaliser la détection des aspects abordés de manière non-supervisée.

Pour finir, nous discuterons des intérêts et des limites de cette approche. Nous évoquerons des pistes et des suggestions d'amélioration dans le but d'affiner cette méthode. Par ailleurs, nous montrerons un exemple de produit employant cette méthode afin d'illustrer un cas d'usage.

# Lexique

- **Embedding** : représentation vectorielle d'un token (généralement d'un mot). Il s'agit d'une manière de représenter une entité textuelle sous la forme d'un vecteur.
- **N-gramme** : il s'agit d'une séquence de N entités textuelles consécutives. Les N-grammes sont généralement évoqués pour parler des expressions récurrentes contenant plusieurs mots tel que « New York City ». On parle de **bigrammes** et de **trigrammes** pour évoquer les 2-grammes et 3-grammes.
- **Bag-of-words** : représentation vectorielle d'un document en indiquant par 1, la présence d'un terme et par 0, l'absence d'un terme. Cette représentation se base sur une liste de termes dont l'ordre définit à quel moment placer un 1 ou 0.
- **Corpus** : ensemble de documents.
- **Document** : une séquence de mots formant une unité textuelle. Il peut s'agir d'un commentaire, d'un paragraphe ou encore d'une phrase par exemple.
- **Token** : un anglicisme pour signifier « jeton ». Il s'agit une unité textuelle qui sert généralement à qualifier un élément que l'on souhaite représenter par un embedding.
- **Sentiment / Opinion** : caractéristique d'un avis qui peut être positif ou négatif. La littérature ajoute parfois le label neutre comme sentiment.
- **Polarité** : valeur estimant le sentiment d'un texte. Ici, une valeur proche de 1 désigne un texte positif tandis qu'une valeur proche de 0 désigne un texte négatif.
- **Aspect** : désigne une caractéristique concrète d'une idée, concept ou d'un domaine. Par exemple, les aspects d'un restaurant peuvent être le service, les prix, l'ambiance, la qualité de la nourriture.
- **Topic** : désigne un thème abstrait d'un corpus. Il est dépendant du corpus analysé.
- **POS tagging** : anglicisme désignant l'étiquetage morpho-syntaxique (part-of-speech tagging). Processus associant les informations grammaticales aux mots d'un texte.
- **Epochs** : dans le contexte de l'apprentissage profond où un modèle est entraîné sur un jeu de données de manière itérative, un epoch correspond à une itération du modèle sur le jeu de données entier.
- **Gensim** : librairie Python spécialisée dans la recherche de thèmes.
- **Tensorflow** : outil destiné à l'apprentissage automatique.
- **Keras** : librairie Python pour l'apprentissage profond

- **Librairie** : un ensemble de fonctionnalités pré-implémentées pour la programmation, codé pour être utilisé facilement pour d'autres développeurs.
- **TALN** : Traitement automatique du langage naturel.

# Table des matières

**Table des figures**

**Liste des tableaux**

**Introduction** **1**

**1 Problématique** **2**

1.1 L'analyse des sentiments et ses limites . . . 2

1.2 L'analyse des sentiments par aspect . . . 2

1.3 Les apports du SemEval . . . 3

1.4 Enoncé de la problématique . . . 4

**2 Motivations du choix du sujet** **4**

**3 Etat-de-l'art** **5**

3.1 Des modèles de word embedding . . . 5

3.1.1 Word2Vec . . . 5

3.1.2 GloVe . . . 6

3.1.3 fastText . . . 6

3.2 Des approches supervisées . . . 7

3.2.1 La détection des aspects via des mesures de similarité . . . 9

3.3 Le topic modeling dans l'analyse des sentiments par aspect . . . 13

3.3.1 L'analyse des sentiments . . . 17

3.4 Conclusion sur l'état-de-l'art . . . 18

**4 Approches et expérimentations** **19**

4.1 Le périmètre du projet . . . 19

4.1.1 Le choix du domaine . . . 19

4.1.2 Le choix des jeux de données . . . 19

4.1.3 Séparation des jeux de données . . . 20

4.2 Matériel . . . 21

4.3 Mise en qualité du corpus . . . 21

4.3.1 Segmentation des commentaires en phrases . . . 21

4.3.2 Nettoyage du texte . . . 23

4.3.3 Filtrage du texte . . . 23

4.4 Analyse exploratoire des commentaires . . . . . 24
4.5 Hypothèses . . . . . 28
4.6 Introduction de l'approche . . . . . 28
4.6.1 Explication du système général . . . . . 28
4.6.2 Modèle de représentations vectorielles utilisé . . . . . 29
4.7 L'analyse des sentiments . . . . . 31
4.7.1 Modélisation du problème . . . . . 31
4.7.2 Métrique utilisée . . . . . 32
4.7.3 Génération de données . . . . . 32
4.7.4 Modèle et résultats . . . . . 35
4.7.5 Interpolation de la polarité . . . . . 38
4.7.6 Analyse des erreurs et résultats . . . . . 40
4.8 Détection des aspects avec des mesures de similarité . . . . . 43
4.8.1 Objectifs . . . . . 43
4.8.2 Méthodologie . . . . . 43
4.8.3 Adapter un algorithme existant . . . . . 44
4.8.4 Détection de la présence ou de l'absence d'aspect . . . . . 45
4.8.5 Recherche de candidats . . . . . 45
4.8.6 Similarité avec les labels . . . . . 46
4.8.7 Gestion des termes peu pertinents . . . . . 48
4.8.8 Expérimentations avec les améliorations potentielles citées . . . . . 49
4.8.9 Algorithme retenu . . . . . 50
4.8.10 Résultats et analyse des erreurs . . . . . 53

**5 Applicabilité de la solution** **54**
5.1 Démonstration d'un produit . . . . . 54
5.2 Généralisation . . . . . 56

**6 Discussion sur les intérêts, limites et pistes d'amélioration** **57**

**Conclusion** **60**

**References** **63**

**Annexe** **64**

# Table des figures


1 Résultat possible issu d'une analyse des sentiments résumant un avis. . . . . 3
2 Architectures en CBOW et SkipGram. . . . . 6
3 Architecture convolutif pour la classification de catégorie. . . . . 8
4 Architecture proposée par Schmitt et al. 2018 . . . . . 9
5 Illustration de l'attention par CAt . . . . . 11
6 Système proposé par Vargas, Pessutto et Moreira 2020 . . . . . 12
7 Système proposé par García-Pablos, Cuadros et Rigau 2017 . . . . . 15
8 Schéma montrant la séparation des aspect-terms et opinion-words. . . . . 16
9 Architecture d'un VAE . . . . . 17
10 Architecture d'un VAE proposé par Hoang, Le et Quan 2019 . . . . . 17
11 Illustration du processus de la Double Propagation . . . . . 18
12 Exemple de commentaire tiré de Citysearch . . . . . 20
13 Schéma synthétisant la segmentation des jeux de données . . . . . 21
14 Exemple de segmentation en phrases . . . . . 22
15 Résultat de la segmentation d'une phrase test avec différentes libraires spécialisées pour comparaison. . . . . 22
16 Exemple de nettoyage de texte. . . . . 23
17 Exemple de filtrage des textes. . . . . 24
18 Répartition du nombre de mots après nettoyage. . . . . 24
19 Répartition des aspects prédéfinis. . . . . 25
20 Nuages de mots par aspect . . . . . 26
21 Distribution du nombre d'aspects et du nombre de sentiments . . . . . 27
22 Répartition du nombre d'aspects parmi les phrases à deux sentiments . . . . 27
23 Échantillon de phrases possédant deux sentiments . . . . . 27
24 Schéma de l'approche . . . . . 29
25 Représentation vectorielle des mots projetés en deux dimensions. . . . . 30
26 Schéma représentant la segmentation d'une phrase. . . . . 31
27 Exemple de séquence d'entrée pour l'apprentissage de la segmentation. . . . 32
28 Échantillon de commentaires de Yelp classés selon leur note. . . . . 33
29 Provenances des données pour la segmentation et filtres appliqués. . . . . 34
30 Schéma de la génération de séquences de termes labélisés en sentiment. . . . 34
31 Illustration d'un BiLSTM. . . . . 36

32 Architecture du réseau BiLSTM utilisée . . . . . 37
33 Evolution du taux de bonnes réponses du modèle BiLSTM . . . . . 38
34 Evolution des coûts du modèle BiLSTM . . . . . 38
35 Instance du problème d'interpolation . . . . . 39
36 Exemple d'interpolation sur un seul terme . . . . . 39
37 Exemple d'interpolation sur le terme "and" . . . . . 39
38 Le déroulé de l'algorithme d'interpolation . . . . . 40
39 Résultat final de la segmentation en sentiments . . . . . 40
40 Prédiction des sentiments sur les données de test . . . . . 41
41 Échantillon d'erreur sur les sentiments . . . . . 42
42 Matrice de confusion du modèle BiLSTM sur les sentiments . . . . . 42
43 Exemple de détection d'aspects sur une phrase avec deux sentiments . . . . 43
44 Proportion des aspects parmi les candidats . . . . . 46
45 Exemple de 30 candidats générés . . . . . 46
46 Représentation des aspects et de la phrases test en deux dimensions. Ce plan est l'espace de projection qui résulte d'une analyse par composantes principale sur un sous-ensemble de mots (en utilisant leur représentation vectorielle donnée par le modèle Word2Vec). . . . . 47
47 Exemple de thésaurus pour les aspects. . . . . 47
48 Représentation des aspects et de la phrases test en deux dimensions . . . . . 48
49 Évolution du f1-score face au nombre de termes $k$ considérés dans la classification en aspects . . . . . 50
50 Processus de détection d'aspect. . . . . 52
51 Évolution du F1-score dans la détection d'aspects . . . . . 53
52 Échantillon de faux positifs dans la détection d'aspects . . . . . 54
53 Exemple montrant le déroulé du processus de synthèse . . . . . 55
54 Schéma d'un produit utilisant notre algorithme . . . . . 55
55 Tableau de bord réalisé synthétisant des données de Yelp . . . . . 56
56 Illustration du concept de la back translation sur une phrase . . . . . 59
57 Architecture BiLSTM utilisée dans la segmentation des sentiments . . . . . 64

# Liste des tableaux


1 Exemple de thésaurus . . . . . 10
2 Liens vers les sources de données . . . . . 20
3 Caractéristiques des données d'entraînement du Word2Vec . . . . . 29
4 Paramètres d'entraînement du modèle Word2Vec . . . . . 29
5 Similarités entre des paires de mots . . . . . 31
6 Hyperparamètres de l'entraînement du réseau BiLSTM . . . . . 36
7 Récapitulatif des scores engendrés par les améliorations suggérée pour CAt. 49
8 Performance de la détection d'aspects à un seuil $p = 0.5$ . . . . . 54

# Introduction

L'ère de l'information a exercé une forte influence dans le développement de nos technologies. L'arrivée du Web 2.0 a donné la possibilité aux internautes d'échanger via diverses plateformes de communication. Ces échanges n'ont cessé d'augmenter donnant lieu à une explosion de la quantité d'information. Aujourd'hui, il est question de tirer profit de ces informations et moyens de communication, notamment en exploitant du contenu généré par les utilisateurs (CGU ou encore user-generated content). Ces derniers représentent le contenu publié par des utilisateurs et contrastent avec les publications d'une entreprise ou d'un éditeur de site web. Les CGU peuvent prendre la forme de commentaires, d'articles, d'avis ou encore des évaluations de commerces et renferment dans certains cas l'opinion des utilisateurs. Ces données deviennent donc intéressantes pour des entités souhaitant améliorer leur image et font l'objet d'étude de plusieurs cas d'usage. À titre d'exemple, certaines marques de luxe proposant un marché en ligne accumulent des commentaires qui serviront à estimer le niveau de satisfaction décliné sous différents angles (éléments négatifs sur le produit, le service, le rapport qualité/prix...) ce qui permettra d'améliorer sa stratégie de commerce. D'autres entreprises utilisent les commentaires publics sur des réseaux sociaux afin d'évaluer l'impression du public sur un évènement organisé. C'est le cas pour la convention Intel AI development en 2018 dont les retours étaient extraits des réseaux de Twitter pendant et après l'évènement. Des analyses ont permis de catégoriser les commentaires et distinguer les thématiques abordées ce qui a permis aux organisateurs d'en déduire des axes d'amélioration pour les futures conventions.

Les CGU deviennent une source riche d'information, car ils peuvent renfermer l'opinion exprimée des individus. En revanche, la grande quantité de données à traiter peut rendre toute analyse manuelle rébarbative et chronophage. C'est pourquoi, il sera question dans ce mémoire de proposer un moyen d'automatiser l'analyse textuelle afin d'établir une synthèse pertinente des commentaires reçus dans un domaine particulier. Cela aura pour but d'obtenir une vue des données textuelles résumant les points importants au niveau de l'opinion qui a été exprimé.

**L'objectif de ce mémoire est donc de montrer comment synthétiser l'opinion exprimée au sein d'un corpus en réalisant une analyse des sentiments par aspect.**

# 1 Problématique

## 1.1 L'analyse des sentiments et ses limites

Afin d'automatiser l'analyse du texte, les méthodes basées sur le TALN deviennent des solutions intéressantes. Un champ d'étude classique et répandu de ce domaine est l'analyse des sentiments dont l'objectif consiste à déterminer de manière automatique la polarité d'un texte. Très généralement, la tâche se réduit à attribuer une étiquette parmi les trois suivantes : « positif », « négatif » ou « neutre » et apparaît comme un problème de classification.

Des méthodes basées sur le champ lexical ou encore sur l'apprentissage profond ont permis d'aborder le problème. Néanmoins, la modélisation de la tâche en une classification de textes présente ses limites. La possibilité de pouvoir retrouver les points abordés dans le texte sort du cadre de la tâche. Il existe, pour certains modèles en apprentissage supervisé, des techniques de visualisation pour mieux appréhender ses résultats (Zhouhan et al. 2017) en tentant de justifier les passages responsables de la décision mais rien ne garantit que ces éléments soient pertinents. De plus, un avis peut être partagé en comportant un avis positif sur un premier aspect et négatif sur un deuxième. Ceci n'est pas traité par les modèles de classification de commentaires.

## 1.2 L'analyse des sentiments par aspect

L'analyse des sentiments par aspect a été introduite dans le but de pouvoir segmenter le sentiment exprimé dans un texte selon des aspects généralement prédéfinis comme le montre la figure 1. Ce champ d'étude permet d'affiner la classification des commentaires en identifiant les aspects abordés puis en déterminant le sentiment par rapport à ces aspects.

L'analyse des sentiments par aspect s'est popularisée par la reprise du problème dans le séminaire du SemEval (Semantic Evaluation). Ce séminaire, basé sur le traitement automatique du langage naturel et sur la recherche, a pour but de faire évoluer et comparer des systèmes récents sur l'analyse sémantique du texte. Il s'agit d'un évènement annuel dans lequel une série de problèmes complexes est posée. Des jeux de données annotées sont mis à disposition pour la résolution de ces défis.

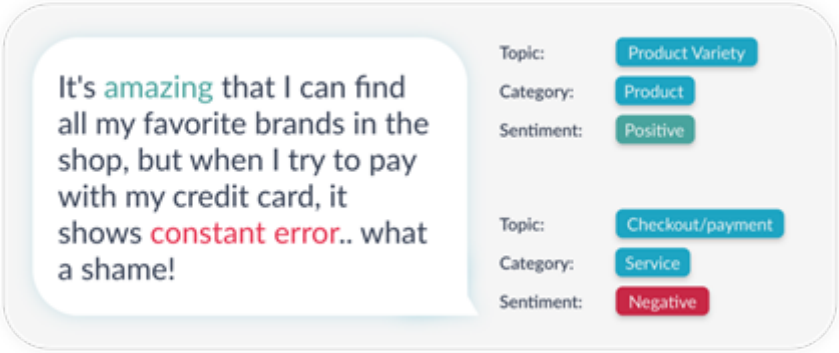


FIGURE 1 – Résultat possible issu d'une analyse des sentiments résumant un avis.
Source : https ://www.symanto.com/api/aspect-based-sentiment/

## 1.3 Les apports du SemEval

Le SemEval a introduit l'analyse des sentiments par aspect pour la première fois en 2014 en proposant deux jeux de données en anglais reprenant des avis sur des restaurants et sur des ordinateurs portables. Les organisateurs ont proposé de séparer le problème en quatre tâches :

- **Aspect term extraction** : identifier les termes référant à un aspect du domaine.
- **Aspect term polarity** : rechercher le sentiment exprimé par ce terme.
- **Aspect category detection** : identifier les catégories d'aspects mentionnés.
- **Aspect category polarity** : déterminer la polarité des catégories.

Dans le cadre de ce mémoire, la problématique revient à résoudre la détection de catégories et la reconnaissance de la polarité pour une catégorie qu'on appellera "aspect".

Ce problème a été repris durant trois années consécutives (de 2014 à 2016). Chaque année a apporté une extension au problème. Une nouvelle formulation a été proposée en 2015 afin de faciliter le problème en reprenant les tâches suivantes :

- **Aspect Category Detection** : pour identifier les catégories d'aspects mentionnés.
- **Opinion Target Expression** : extraire les termes référant à un aspect du domaine.
- **Sentiment Polarity** : déterminer le sentiment d'un couple catégorie-terme.

En 2016, la tâche a été étendue pour adopter un point de vue multilingue en introduisant d'autres jeux de données dans diverses langues, notamment en français, arabe et néerlandais.

## 1.4 Enoncé de la problématique

Dans la volonté d'affiner l'analyse des sentiments, il sera question de modéliser un algorithme visant à produire une synthèse des aspects abordés et des opinions exprimées. Des organisations amassent une grande quantité d'avis à traiter, rédigés par leurs clients ou par des utilisateurs de leurs services ou produits. **L'objectif est de proposer un moyen d'automatiser le traitement des avis dans un domaine particulier.** Il sera préférable de traiter un domaine à la fois (commentaires sur des restaurants, avis sur des produits électroménagers...). Les commentaires textuels adressés à un domaine spécifique formeront le corpus à analyser. Dans la littérature, un aspect peut désigner un terme du texte faisant référence à une caractéristique du domaine (restaurants : service, cuisine, ambiance...) ou encore à une catégorie d'aspects (les caractéristiques). Ici, les aspects se réfèreront aux catégories.

En considérant un corpus issu d'un domaine, **l'enjeu est de pouvoir offrir une visualisation simple des sentiments par aspect qui servira de synthèse du corpus.** Cela permettra d'avoir une vue instantanée, résumant les points essentiels du corpus. Ainsi, nous allons nous servir d'une analyse des sentiments par aspect.

En tenant compte de la tâche chronophage d'annotation des données, le travail de synthèse aura pour contrainte d'éviter la labélisation manuelle des données. Nous supposerons que les avis ne sont pas annotés par aspect. Par conséquent, **on considérera la reconnaissance des aspects comme une tâche non-supervisée**, voire faiblement supervisée dans le sens où chaque document n'est pas annoté avec un ou plusieurs aspects. En revanche, les aspects peuvent être choisis à l'avance pour venir préciser le résultat.

De ce fait, il sera question de traiter automatiquement les avis reçus afin de ressortir les proportions positif/négatif pour chaque aspect sans données annotées en aspect.

# 2 Motivations du choix du sujet

Le choix de la problématique a été inspiré par les missions qui m'ont été proposées dans le cadre de ma formation d'apprenti au sein du Ministère de l'Économie, des Finances et de la Relance. Plusieurs directions ont récolté des verbatims vis-à-vis d'un service ou encore d'une thématique définie (équipements, site web...). La quantité de commentaires rend l'analyse manuelle difficile à pratiquer. De plus, classifier les commentaires est peu utile car cela ne permet pas de comprendre les éléments positifs ou négatifs au sein des textes ;

des analyses doivent être faites en amont. Il est donc intéressant d'avoir un algorithme pour offrir une vue sur le sentiment par aspect.

D'un point de vue personnel, il est peu commun de posséder des données pré-annotées en aspects. De ce fait, adopter une méthode non-supervisée apporte une solution pragmatique. Cela donne la possibilité de choisir les aspects pertinents.

# 3 Etat-de-l'art

## 3.1 Des modèles de word embedding

Dans un premier temps, nous avons besoin de comprendre comment représenter le texte de manière numérique. Pour cela, nous allons expliciter la notion d'embedding. L'embedding d'un mot est une représentation vectorielle du mot lui-même. Ici, nous présentons brièvement les méthodes d'embedding qui seront mentionnées.

### 3.1.1 Word2Vec

Le Word2Vec de (Mikolov et al. 2013) permet de produire une représentation vectorielle continue d'un mot à partir d'un corpus. L'enjeu est d'associer un mot à son contexte (défini par les mots qui l'avoisinent à un instant donné). Une fenêtre est définie autour du mot qui délimite son contexte. Chaque mot est représenté dans sa forme en Bag-of-words qui, avec leur contexte respectif, forme la donnée d'entraînement. Afin d'avoir une représentation continue, ces données sont utilisées pour entraîner un modèle neuronal sur une tâche prétexte. Deux tâches donnant deux architectures neuronales peuvent être envisagées (illustrées sur la Figure 2).

- Le Skip-gram est une architecture pour la prédiction d'un contexte à partir d'un mot.
- L'architecture en CBOW a pour but de prédire un mot étant donnée un contexte.

Les fonctions de coûts sont définies de la manière suivante :

$$L_{CBOW} = \frac{1}{M}\sum_{i=1}^{M} log(p(w_i|w_{c(i)})) \tag{1}$$

$$L_{sg} = \frac{1}{M}\sum_{i=1}^{M}\sum_{-k\leq j\leq k, j\neq 0} log(p(w_{i+j}|w_i)) \tag{2}$$

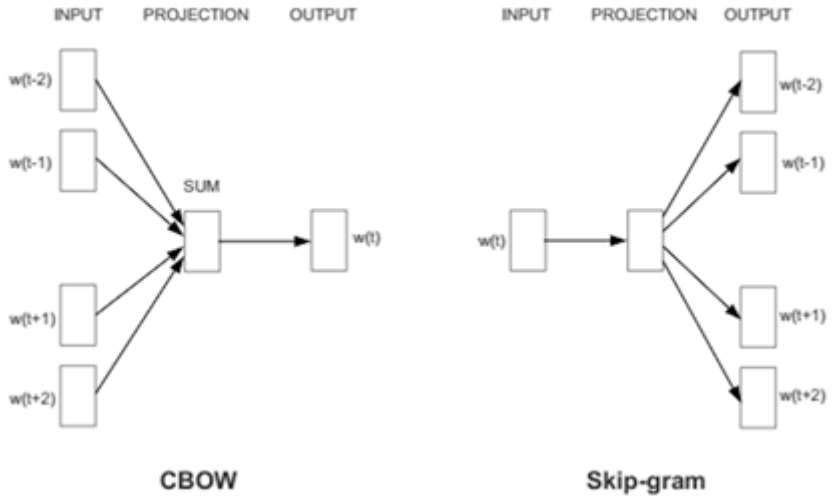


FIGURE 2 – Architectures en CBOW et SkipGram.
Source : https ://israelg99.github.io/

où $M$ désigne la taille des données, $c(i)$ désigne le contexte du i-ème mot et la probabilité conditionnelle définit par $p(w_i|w_j) = \frac{exp(w_i^T w_j)}{\sum_{w \in V} exp(w^T w_i)}$ pour $V$ le vocabulaire.

L'embedding du mot $w_i$ peut être extrait du réseau de neurones en récupérant les i-èmes poids de chaque neurone de la première couche en Skip-gram (et dernière en CBOW).

### 3.1.2 GloVe

GloVe de PENNINGTON, SOCHER et MANNING 2014 se base sur la co-occurrence des mots en définissant la matrice associée $X$ où $X_{ij}$ est la fréquence d'apparition du mot $i$ dans le contexte de $j$. Une taille de fenêtre est fixée. Afin que deux mots $i$ et $j$ soient proches dans l'espace d'encodage lorsque leur co-occurrence est élevée, nous aimerions avoir $w_i^T w_j + b_i + b_j = log(X_{ij})$ donnant la fonction de coûts suivante :

$$L = \sum_{i=1}^{M} \sum_{j=1}^{M} f(X_{ij})(w_i^T w_j + j_i + b_j - log(X_{ij}))^2 \tag{3}$$

$$f(X_{ij}) = \begin{cases} (\frac{X_{ij}}{x_{max}})^a & \text{si } X_{ij} < b_i \text{ et } b_i \text{ le biais associé au mot } i \text{ avec } a \text{ un hyperparamètre.} \\ 1 & \text{sinon} \end{cases} \tag{4}$$

### 3.1.3 fastText

FastText de BOJANOWSKI et al. 2017 est assez similaire au Word2Vec à l'exception que la représentation d'un mot se fait à partir des n-grammes de caractères qui le composent.

Par exemple, le mot « words » se décomposerait en <wor, ord, rds> pour n=3. Il est possible d'obtenir des vecteurs pour chaque n-gramme par l'intermédiaire d'une architecture en Skip-gram. L'embedding d'un mot est la somme des n-grammes qui le constituent. La décomposition d'un mot en n-grammes permet de générer des embeddings pour des mots n'appartenant pas au corpus d'entrainement sous réserve que les n-grammes qui le composent sont connus.

## 3.2 Des approches supervisées

Bien que la problématique suggère une approche non-supervisée, cette section présente des approches importantes parmi les solutions supervisées et contiennent des directions intéressantes pour la suite.

Les défis lancés par le SemEval ont permis d'apporter des solutions intéressantes dans l'analyse des sentiments par aspect. En 2016, il a été montré que les meilleures méthodes se basaient sur des représentations vectorielles continues des termes.

Toh et Su 2016 ont traité la classification des phrases dans différentes catégories d'aspects via des classifieurs binaires en one-versus-all. Les auteurs exploitent les caractéristiques de chaque mot (appartenance à un cluster...) afin de représenter une phrase comme une séquence de vecteurs. Chaque vecteur est la représentation d'un mot. La séquence constitue l'entrée d'un réseau convolutif sortant une distribution sur les catégories (voir Figure 3).

De même, Xenos et al. 2016 ont basé leur solution sur une représentation des mots par Word2Vec et des méthodes ensemblistes fondées sur des SVM pour la classification d'aspects et l'estimation de la polarité. Néanmoins, ces méthodes sont connues pour être sensibles aux termes peu pertinents.

Lee et al. 2017 proposent un autre paradigme en cherchant à extraire les séquences de mots qui font référence à un aspect (aspect terms) pour les relier à la catégorie correspondante. Schmitt et al. 2018 argumentent que cette méthode pose un problème lorsqu'une catégorie n'est pas nécessairement reliée à un passage du texte mais peut être référée de manière implicite. Pour contourner ce problème, Schmitt et al. 2018 utilisent un réseau convolutif à plusieurs sorties. Chaque sortie s'occupe d'estimer la polarité d'une catégorie en calculant une probabilité pour chaque sentiment. Si chaque polarité est inférieure à 0.5, le texte n'est pas associé à une catégorie (par supposition). Ainsi, cette méthode repose sur

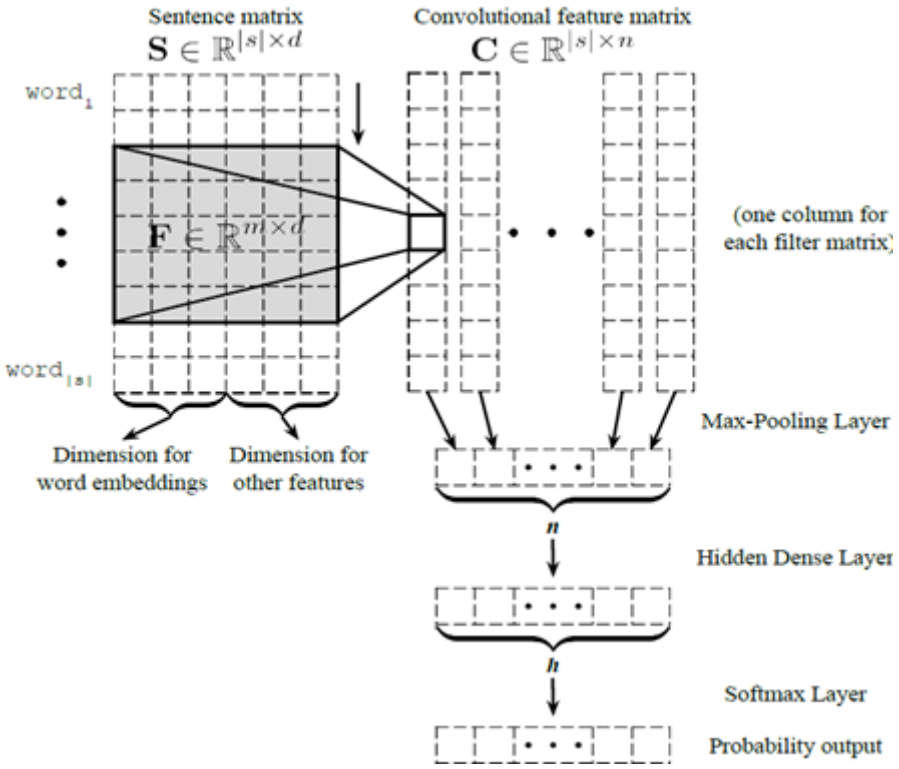


FIGURE 3 – Architecture convolutif pour la classification de catégorie.
Source : TOH et SU 2016

un unique modèle (voir Figure 4).

Ce type d'architecture a gagné de l'intérêt dans la détection de catégories. MOVAHEDI et al. 2019 utilisent une architecture similaire et introduisent une nouvelle couche : le Topic-Attention Layer, dont le but est d'accorder plus d'importance aux tokens responsables d'une catégorie en se basant sur le mécanisme d'attention de BAHDANAU, CHO et BENGIO 2016. L'attention consiste à se focaliser sur certaines parties d'une séquence en attribuant un score d'attention (d'importance) à chaque token. Cela permet de neutraliser les éléments parasites et de conserver les plus importants pour la tâche.

Ainsi, plusieurs Topic-Attention Layer sont mises en parallèle, chacune responsable d'une catégorie. Chaque sortie détermine si une catégorie est présente. Afin d'encourager la distinction des catégories, les auteurs introduisent un terme de régularisation pour renforcer l'orthogonalité des poids $T_n$ à la dernière couche. La fonction de coût [5] est donc la somme entre la fonction $J$ pour corriger les prédictions de catégories et la régularisation $U$ [6]. $\theta$ représente les paramètres du modèles.

$$L(\theta) = J(\theta) + U(\theta) \tag{5}$$

$$U(\theta) = ||T_n T_n^T - I|| \tag{6}$$

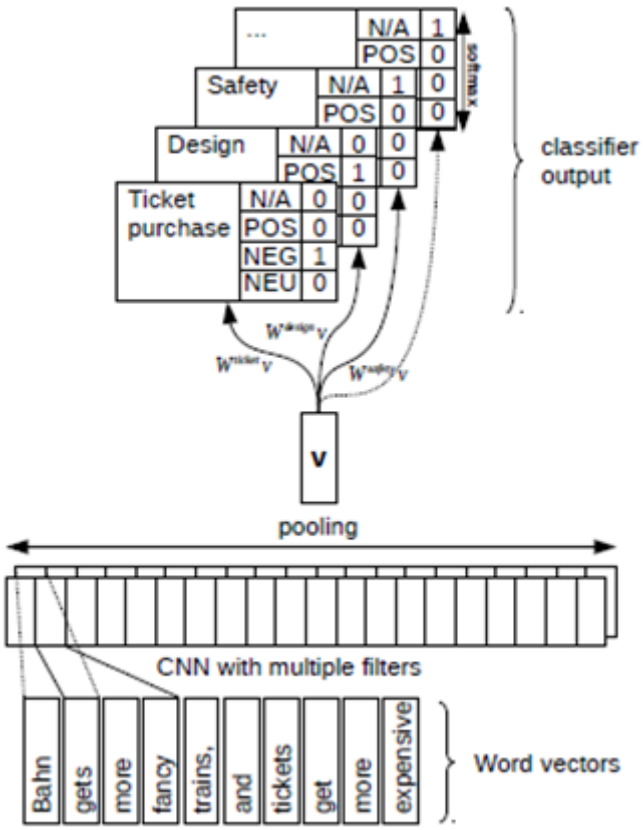


FIGURE 4 – Architecture proposée par SCHMITT et al. 2018
Source : SCHMITT et al. 2018

Afin de mieux isoler les aspects, NI, LI et MCAULEY 2019 suggèrent de segmenter le texte en EDU (Element Discourse Unit). Le but est de séparer les différents faits et jugements abordés (EDU) ce qui améliorerait les performances par rapport à une segmentation en phrases.

### 3.2.1 La détection des aspects via des mesures de similarité

Face à la quantité volumineuse de données à traiter, certaines études ont cherché des méthodes qui ne demandent pas d'annotation mais seulement la liste des aspects à reconnaitre, rendant la tâche faiblement supervisée. Ici, les mesures de performances restent identiques aux méthodes supervisées puisque les catégories sont connues. Afin de détecter un aspect, des méthodes se basent sur le calcul de similarité entre un document et un aspect. GHADERY et al. 2019 proposent une mesure de similarité basés sur le soft-cosine de SIDOROV et al. 2014 définie ci-dessous et le regroupement des textes en clusters. Idéalement, des textes au sein d'un même cluster partageraient des informations similaires, d'où l'intérêt d'exploiter un clustering.

| **Aspects** | **Seed words** ($s = s_1, s_2...$) |
|---|---|
| **Service** | Service, staff, friendly, attentive, manager |
| **Price** | Price, cheap, expensive, money, affordable |

TABLE 1 – Exemple de thésaurus

$$softcos(a, b) = \frac{\sum_{i,j} s_{ij} a_i b_j}{\sqrt{\sum_{i,j} s_{ij} a_i a_j} \sqrt{\sum_{i,j} s_{ij} b_i b_j}} \tag{7}$$

avec $s_{ij} = sim(f_i, f_j)$ et $f_i$ les caractéristiques d'un mot $i$ et $sim$ une mesure de similarité.

Ici, un document représente une phrase. Elle aborde un aspect si la similarité entre cette phrase et l'aspect dépasse un certain seuil. Les aspects sont qualifiés par des listes de mots manuellement instanciée formant un **thésaurus** (voir le tableau 1).

Un clustering des documents est effectué sur un jeu de données annexe (portant sur un domaine très proche du jeu de données à étudier) par le moyen d'un KMeans en utilisant des embeddings Word2Vec.

La similarité entre une phrase x et un aspect $a_i$ est définie comme suit (avec $\alpha$ un hyper-paramètre) :

$$score_{a_i}(x) = \alpha SentScore_{a_i}(x) + (1 - \alpha) ClustScore_{a_i}(centroid(cluster(x))) \tag{8}$$

$$SentScore_{a_i}(x) = \frac{exp(SentSim_{a_i}(x))}{1 + exp(SentSim_{a_i}(x))} \tag{9}$$

$$SentSim_{a_i}(x) = \frac{1}{|s|} \sum_{j=1}^{|s|} softcos(x, s_j) \tag{10}$$

$$ClustScore_{a_i}(c_k) = \frac{exp(ClustSim_{a_i}(c_k))}{1 + exp(1 + exp(ClustSim_{a_i}(c_k)))} \tag{11}$$

$$ClustSim_{a_i}(c_k) = \frac{1}{|c_k|} \sum_{x \in clust(c_k)} SentSim_{a_i}(x) \tag{12}$$

où $clust(c_k)$ := cluster ayant pour centroïde $c_k$.

Ainsi, un document est associé à une ou plusieurs catégories dès lors que la similarité entre le document et une catégorie dépasse un certain seuil. Ce seuil doit être fixé à l'avance.

L'approche de TULKENS et CRANENBURGH 2020 a pour but d'assigner une catégorie à

chaque phrase d'un document. Le procédé se compose de trois étapes :

- **Extraction des termes relatifs aux aspects** : Les auteurs ont observé que les noms communs les plus fréquents sont généralement des candidats raisonnables pour être vu comme aspect terms. Ainsi, un POS-tagger est utilisé pour retrouver ces termes.
- **Sélection d'aspects** : Le but ici est de produire une représentation vectorielle d'un document. Les auteurs introduisent la Contrastive Attention afin de pondérer les mots qui composent un document $S$. Un ensemble de termes de référence est prédéfini dont leur embedding forme une matrice $A$. Ces termes ne sont pas classés par catégories. Les auteurs suggèrent d'obtenir des embeddings à partir d'un corpus appartenant au même domaine que l'ensemble des documents étudiés.

$$att_w = \frac{\sum_{a \in A} rbf(w, a, \gamma)}{\sum_{w' \in S, a \in A} rbf(w', a, \gamma)} \tag{13}$$

$$rbf(x, y, \gamma) = exp(-\gamma ||x - y||_2^2) \tag{14}$$

  Une représentation d'un document est obtenue de la manière suivante : $d = \sum_{w' \in S} att'_w.w'$
- **Désignation des catégories** : Sachant qu'une catégorie est représentée par un mot, il est possible d'avoir un embedding d'une catégorie. Trouver la catégorie qui représente au mieux le document d se calcule ainsi : $y' = \underset{x \in C}{\text{argmax}}\ cos(d, c)$. La similarité cosinus est définie de la manière suivante :

$$cos(u, v) = \frac{u^T v}{||u||.||v||} \tag{15}$$

De cette manière, chaque document est associé à une catégorie parmi celles prédéfinies. Cette méthode a l'avantage d'être interprétable car il est possible de vérifier les termes responsables de la décision grâce aux poids d'attention.

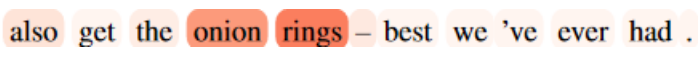


FIGURE 5 – Illustration de l'attention par CAt
L'intensité du rouge indique le poids d'attention donné au mot.
Source : TULKENS et CRANENBURGH 2020

VARGAS, PESSUTTO et MOREIRA 2020 proposent une méthode basée uniquement sur des mesures de similarités. L'idée est toujours de calculer la similarité entre un terme et une catégorie.

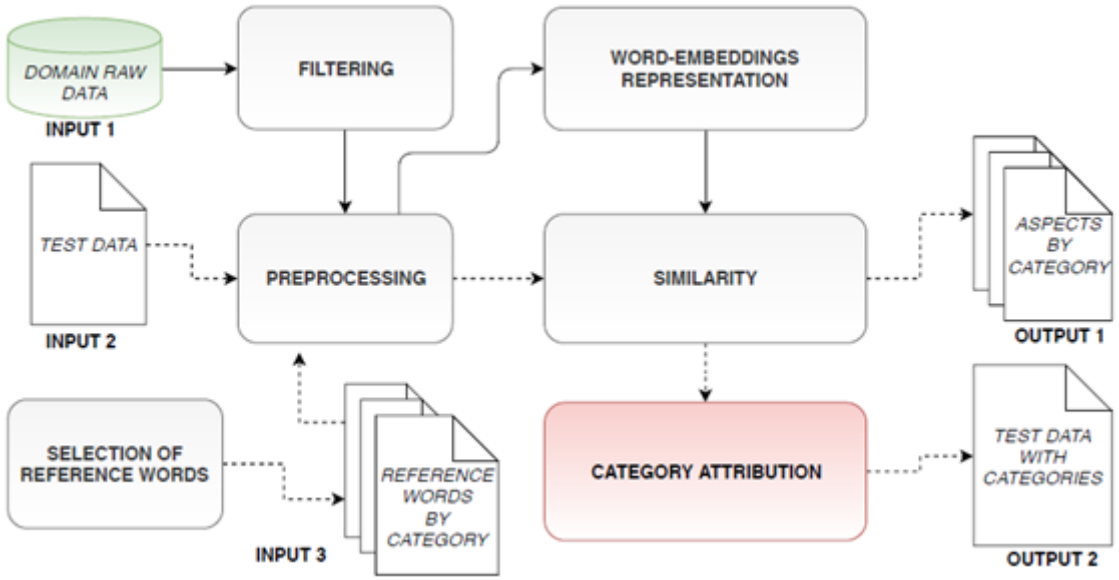


FIGURE 6 – Système proposé par VARGAS, PESSUTTO et MOREIRA 2020
Source : VARGAS, PESSUTTO et MOREIRA 2020

Le système, illustré sur la Figure 6, prend trois entrées :

– l'ensemble des documents,

– un corpus du même domaine que les documents pour la génération d'embeddings,

– un terme pour chaque catégorie.

Le procédé est le suivant :

- Les données sont nettoyées et filtrées afin de garder uniquement les documents relatifs au domaine (input 1) et retirer les termes parasites (stopwords, ...).
- Pour convertir du texte en embedding, les auteurs ont utilisé le Word2Vec.
- Chaque document sera attribué un score de similarité pour chaque catégorie. Pour cela, chaque terme w du document suit ce procédé pour une catégorie donnée :
  - On recherche les n termes les plus similaires à $w$.
  - Pour chaque terme similaire, on calcule la similarité cosinus avec la catégorie et agrège ces valeurs en réalisant une moyenne $a$.
  - On calcule la similarité $b$ entre w et la catégorie.
  - La similarité finale par rapport à la catégorie est la somme $a + b$.
- Ce procédé est répété pour chaque catégorie. Ainsi, pour un terme d'un document, on obtient une liste de scores de similarité pour chaque catégorie. La similarité maximale détermine la catégorie.

- Pour attribuer une catégorie à un document, il suffit de prendre la catégorie majoritairement attribuée aux termes qui le composent.

D'après les auteurs, cette méthode s'inspire du concept d'attention où chaque terme d'un document possède une importance différente vis-à-vis d'une catégorie.

## 3.3 Le topic modeling dans l'analyse des sentiments par aspect

Un topic model (modèle de sujet) est un modèle statistique cherchant à ressortir les topics dans un ensemble de documents. Plusieurs techniques sont au cœur de cette technologie dont la LSA (Latent Sementic Analysis), PLSA (Probabilistic Latent Sementic Analysis) et NMF (Non-Negative Matrix Factorization).

Dans l'analyse des sentiments par aspect, un algorithme régulièrement utilisé est la LDA (Latent Dirichlet Allocation) [Blei, Ng et Jordan 2003]. Il s'agit d'un modèle génératif supposant que chaque document est un mélange de topics latents et chaque topic est défini par une distribution sur le vocabulaire. On suppose que chaque terme d'un document d a été généré de cette façon :

- Tirage d'un topic $z$ suivant une loi multinomiale de paramètre $\theta_d$ (distribution des topics du document $d$) qui est tiré d'une distribution de Dirichlet de paramètre $\alpha$.
- Tirage d'un mot w suivant une loi multinomiale de paramètre $\phi_z$ (une distribution des mots dans le topic $z$) qui est tiré d'une distribution de Dirichlet de paramètre $\beta$.

Les paramètres $\theta$ et $\phi$ regroupent respectivement l'ensemble des distributions de topics et une distribution de mots pour chaque chaque topic. Ces derniers sont les inconnues à retrouver. Pour cela, l'échantillonnage de Gibbs est employé. Il s'agit d'un algorithme itératif qui modélise le problème comme un processus markovien dans lequel les probabilités stationnaires désignent les paramètres recherchés.

La LDA permet de découvrir des thèmes qui peuvent servir d'aspect. Par exemple, Tran, Ba et Huynh 2019 utilisent la LDA à partir d'aspect terms extraits par un modèle pré-entrainé. Cela permet de réduire le bruit contenu dans le texte et d'obtenir des topics de meilleure qualité.

Néanmoins, les topics trouvés par des algorithmes tels que la LDA ne forment pas nécessairement des aspects. En effet, ces modèles tendent à détecter des topics qui segmentent de manière globale les termes d'un document. Ainsi, ces topics ne peuvent se présenter

**Algorithm 1** LDA

Affecter aléatoirement un topic pour chaque mot du vocabulaire.
**for** $k \leftarrow 1$ à $N$ **do**
  **for** $d$ un document **do**
    **for** $w$ un terme de $d$ **do**
      Calculer $\forall i, p(topic_i|w) = p(w|topic_i).p(topic_i|d)$
      Affecter aléatoirement un topic a $w$ en suivant cette distribution.
    **end for**
  **end for**
**end for**

comme des aspects pertinents. Titov et McDonald 2008 introduisent les notions de topics globaux et locaux, où les topics locaux correspondent à des aspects.

Afin de déterminer les topics locaux, Brody et Elhadad 2010 emploient la LDA à l'échelle des phrases et non des documents ou commentaires entiers. Pour déterminer le nombre optimal de topics, les concepts de validation d'un clustering sont empruntés en approchant le problème comme un clustering de phrases.

Pour déterminer le sentiment des aspects trouvés, Brody et Elhadad 2010 se reposent sur les adjectifs employés. Ces derniers sont extraits en considérant les formes négatives et les conjonctions. Par exemple, si le mot « delicious » a été employé dans un contexte négatif, il sera remplacé par le bigramme « not – delicious ». La polarité des aspects est déterminée à partir d'un graphe construit en reliant les adjectifs entre eux. On attribue un sentiment à un sous-ensemble d'adjectifs en repérant des pairs d'adjectifs avec des polarités opposées. Cela est réalisé en remarquant les préfixes tels que « in », « un », « dis », « non ». Par exemple, une paire possible peut être les mots « grateful » et « disgrateful » qui ont bien une polarité opposée. La suite se fait par propagation de labels dans le graphe.

García-Pablos, Cuadros et Rigau 2017 proposent une modification de la LDA (W2VLDA) afin d'introduire la segmentation des sentiments. L'algorithme prend en entrée :

- le corpus de documents
- un thésaurus : liste de termes pour chaque aspect et un terme pour chaque sentiment.

Il renvoie après traitement deux éléments : la liste des mots du corpus segmenté par aspect et sentiment ainsi qu'une segmentation identique à l'échelle des phrases. Le système est schématisé dans la figure 7.

La méthode se compose de deux étapes. La première consiste à obtenir un classifieur qui sépare les termes relatifs aux aspects des termes relatifs aux sentiments (on suppose

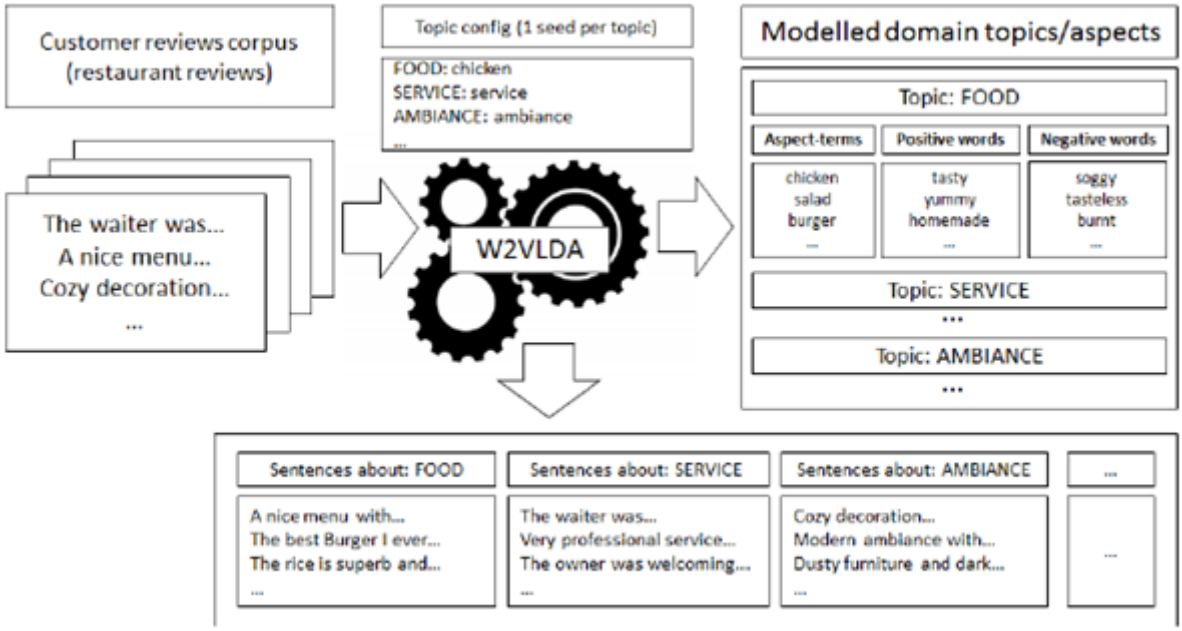


FIGURE 7 – Système proposé par GARCÍA-PABLOS, CUADROS et RIGAU 2017
Source : GARCÍA-PABLOS, CUADROS et RIGAU 2017

qu'il n'y a pas d'autres types). Pour cela, un jeu de données est construit en reprenant les mots du thésaurus en considérant chaque apparition de ces mots dans le corpus avec leur contexte associé (termes environnants). La i-ème entrée est donc $(w_{i-2}, w_{i-1}, w, w_{i+1}, w_{i+2}; label_i)$ où le $label_i$ désigne soit un aspect term ou un mot d'opinion. Chacun de ces termes sont remplacés par leur cluster émit d'un Brown clustering sur un corpus annexe. Enfin, un classifieur est entraîné sur ce jeu de données pour prédire la nature d'un terme. Les probabilités sur les labels en sortie du classifieurs seront utilisées durant la prochaine étape. La méthodologie est schématisée sur la figure 8 :

La seconde étape consiste à déterminer les topics en changeant l'algorithme de la LDA pour prendre en compte le sentiment en plus des topics. On suppose que chaque mot $n$ d'un document $d$ a été généré de la sorte :

- Tirage d'un topic $z_{d,n}$ suivant une loi Multinomiale de paramètre $\theta_d$.
- Tirage de la nature $y_{d,n} \in \{ aspect\ term, opinion\ word \}$ selon une loi $Ber(\pi_{d,n})$
- Si $y_{d,n} = aspect\ term$, tirage d'un mot suivant la distribution de mots $\phi_{z_{d,n},A}$ du topic
- Si $y_{d,n} = opinion\ word$, tirage d'une polarité $v_{d,n} \in \{ positif, negatif \}$
  - Si $v_{d,n} = positif$, tirer un mot suivant la distribution des mots positifs $\phi_{z_{d,n},P}$
  - Si $v_{d,n} = negatif$, tirer un mot suivant la distribution de mots négatifs $\phi_{z_{d,n},N}$

Les topics sont prédéfinis par le thésaurus. Ainsi, il est question de savoir pour un mot donné, quel topic lui associer. Pour cela, les auteurs s'appuient sur la similarité entre les embeddings des termes et ceux du thésaurus. Elle est intervenue pour calculer les probabilités de tirer un topic pour un mot donné, le choix de la polarité et de la nature du mot.

Comme alternative à la LDA, Srivastava et Sutton 2017 suggèrent une autre méthode pour le topic modeling, Autoencoded Variational Inference For Topic Model (AVITM) basée sur un modèle d'auto-encodeur variationnel. Un auto-encodeur (AE) est une architecture en apprentissage profond composé de deux parties :

– un encodeur pour encoder les données dans un espace latent,

– un décodeur pour retrouver la donnée d'origine.

La donnée est encodée en une dimension réduite dans l'idée de compresser pour « synthétiser » la donnée. Un auto-encodeur variationnel (VAE) étend la définition d'un AE dans lequel la donnée encodée est générée à l'aide une distribution aléatoire (gaussienne) dont les paramètres sont donnés par l'encodeur (représenté schématiquement ci-dessous).

Hoang, Le et Quan 2019 ont appliqué ce concept dans l'analyse des sentiments par aspect, pour la recherche de catégories et de sentiments. Une première version propose

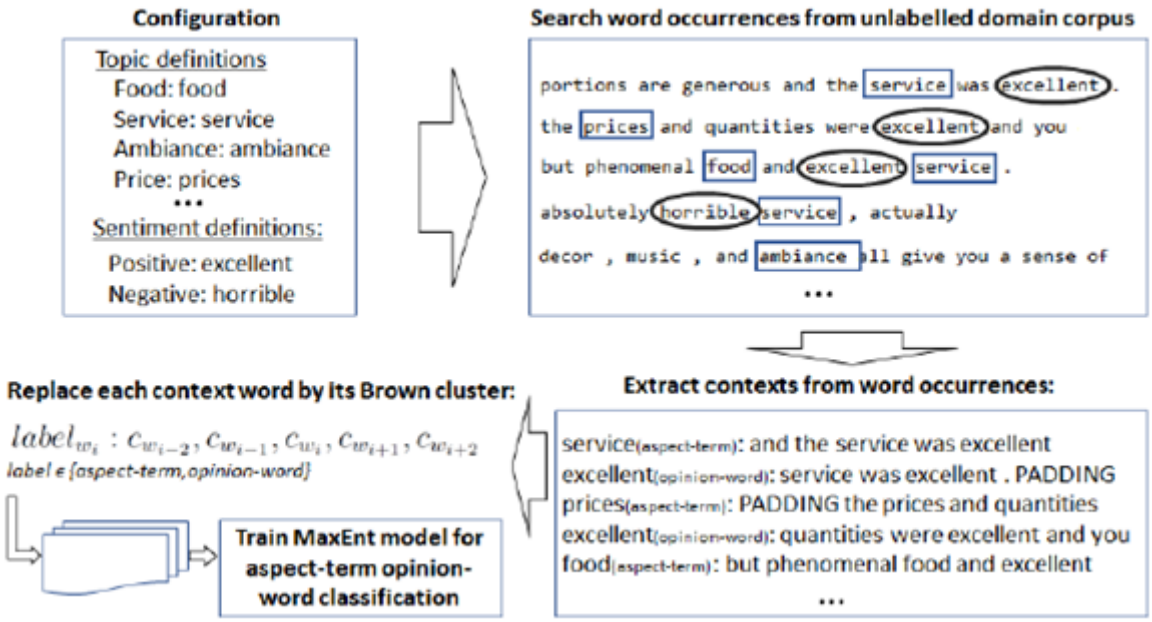


Figure 8 – Schéma montrant la séparation des aspect-terms et opinion-words.
Source : García-Pablos, Cuadros et Rigau 2017

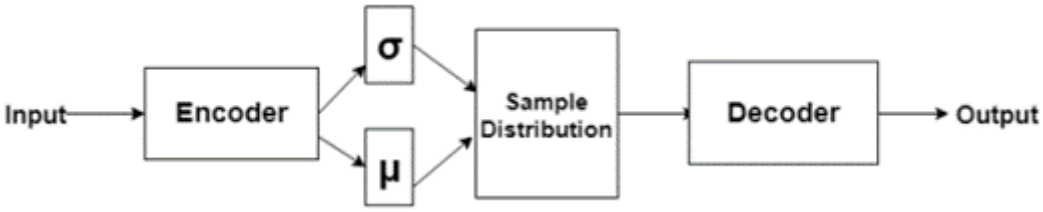


FIGURE 9 – Architecture d'un VAE
Source : https ://theailearner.com/tag/variational-autoencoders/

d'incorporer un thésaurus afin de biaiser les topics vers ces aspects. Cette contrainte est modélisée dans la fonction de coût.

Pour ajouter la séparation des sentiments, les auteurs introduisent un deuxième bloc en parallèle, de la même manière, en incorporant un thésaurus sur les sentiments. L'architecture est donnée dans la figure 10.

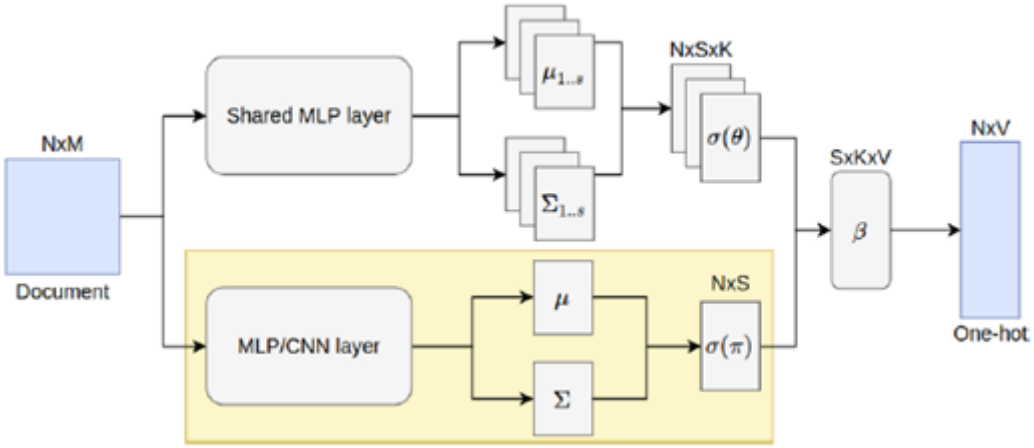


FIGURE 10 – Architecture d'un VAE proposé par HOANG, LE et QUAN 2019
Source : HOANG, LE et QUAN 2019

### 3.3.1 L'analyse des sentiments

L'analyse des sentiments est généralement supervisée avec un corpus où chaque document est annoté avec un sentiment. Un modèle d'apprentissage automatique (SVM, MLP, RNN...) peut être entraîné sur ce corpus afin de classifier de nouveaux documents. Les modèles atteignant les meilleures performances sont généralement issus des modèles de langues tels que BERT (DEVLIN et al. 2019), RoBERTA (LIU et al. 2019), XLNet (YANG et al. 2020), pré-entraînés sur des jeux de données volumineuses [1].

Dans un contexte non-supervisée, il est possible d'utiliser une ressource lexicale qui recense des termes en attribuant un score de polarité tel que SentiWordNet ou Vader. Néan-

1. Source : NLP progress

moins, cela n'aide pas à lever l'ambiguïté des termes (dans le contexte d'une négation par exemple). Des systèmes à base de règles peuvent venir se rajouter pour affiner l'analyse.

L'algorithme de la Double Propagation (Qiu et al. 2011) a pour but d'extraire les termes relatifs aux aspects ainsi que le sentiment associé. L'idée est d'exploiter les relations entre un terme d'opinion et un terme d'aspect : une opinion s'exprime généralement par rapport à un aspect. Le processus est itératif et commence à partir d'un lexique d'opinion, comme illustré sur la Figure 10. Des règles exploitent les dépendances grammaticales reliées à ces termes d'opinion, ce qui permet de trouver les termes d'aspects. Ces derniers sont utilisés de la même manière pour trouver de nouvelles opinions. Ainsi, les lexiques des opinions et des aspects s'agrandissent au fur et à mesure des itérations jusqu'à ce qu'aucun autre mot n'ait été trouvé pour enrichir les lexiques.

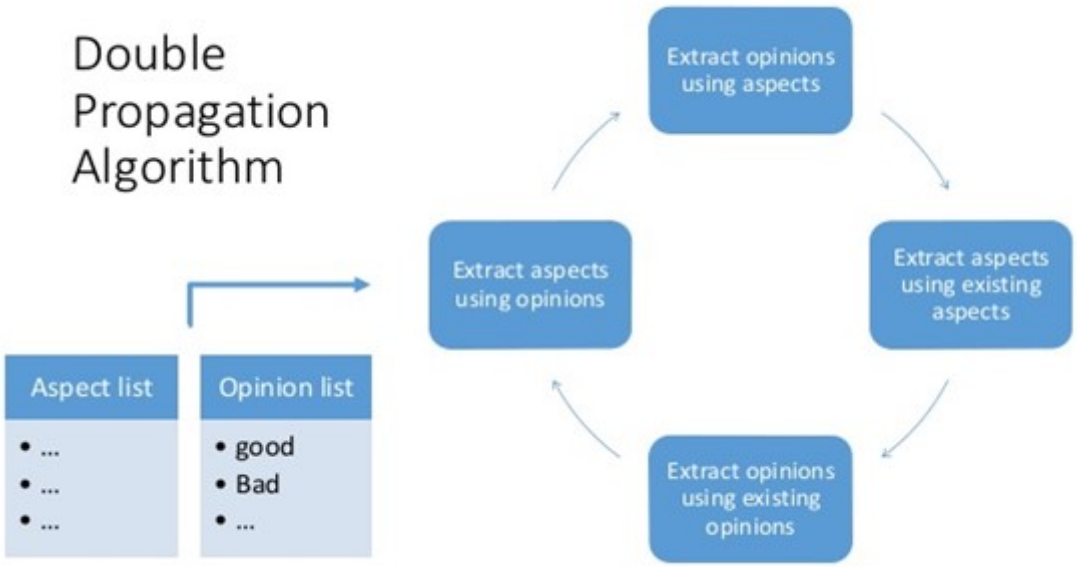


Figure 11 – Illustration du processus de la Double Propagation
Source : Qiu et al. 2011

## 3.4 Conclusion sur l'état-de-l'art

Une difficulté dans l'analyse des sentiments par aspect réside dans le moyen de relier le sentiment à l'aspect. Nous avons vu des méthodes pour extraire l'aspect, d'autres pour estimer le sentiment d'un texte. Néanmoins, les études présentées se concentrent souvent sur une partie indépendamment de l'autre.

Pour concilier la détection d'aspect avec l'analyse des sentiments, Anoop et Asharaf

2020 emploient une méthode de topic modeling dans laquelle est extraite une liste de topics. Ces derniers sont associés, dans la mesure du possible, à un aspect. Les topics sont utilisés pour extraire les phrases faisant référence à au moins un aspect. Par la suite, le sentiment est estimé pour chacun de ces phrases à l'aide d'un modèle de classification supervisé. Cependant, l'association topic-aspect doit se faire manuellement.

D'autres méthodes proposent de trouver les aspects et le sentiment simultanément. L'algorithme de la Double Propagation en est un exemple. Sa nature non-supervisée est un avantage. En revanche, la méthode se base sur la co-occurrence des noms et adjectifs qui ne la rend pas robuste face aux documents courts et ne prend pas compte de la sémantique du texte. La W2VLDA de García-Pablos, Cuadros et Rigau 2017 se range dans la même catégorie. Elle adapte l'algorithme de la LDA pour incorporer l'estimation du sentiment. Les auteurs pointent une faiblesse dans sa capacité à traiter les expressions négatives. Cette méthode nécessite tout de même d'entraîner un classifieur pour distinguer les termes porteurs d'aspect et ceux porteurs de sentiment.

Afin de développer notre approche, **nous allons garder l'estimation du sentiment comme une tâche supervisée.** L'opinion est un concept abstrait dont les méthodes non-supervisées risquent de poser des problèmes de fiabilité. Cependant, nous verrons comment réaliser la détection d'aspect et le concilier à l'analyse des sentiments.

# 4 Approches et expérimentations

## 4.1 Le périmètre du projet

### 4.1.1 Le choix du domaine

Pour simplifier l'étude, les expérimentations seront limitées à un seul domaine. Ainsi, **nous choisirons le domaine de la restauration dans le contexte où des clients peuvent commenter le restaurant** depuis une plateforme en ligne. Les données sont accumulées pour former un corpus qui sera à analyser pour en extraire les aspects et sentiments.

### 4.1.2 Le choix des jeux de données

Afin de mener cette étude, nous allons nous pencher sur deux jeux de données.

- L'échantillon de commentaires introduit à l'occasion du SemEval de 2014 à 2016.

- Un échantillon issu de Citysearch, un guide en ligne fournissant des informations sur différents domaines et activités d'une ville tels que ses commerces, ses activités nocturnes ou encore ses restaurants. Il recense plusieurs millions de commentaires donnés par des internautes.

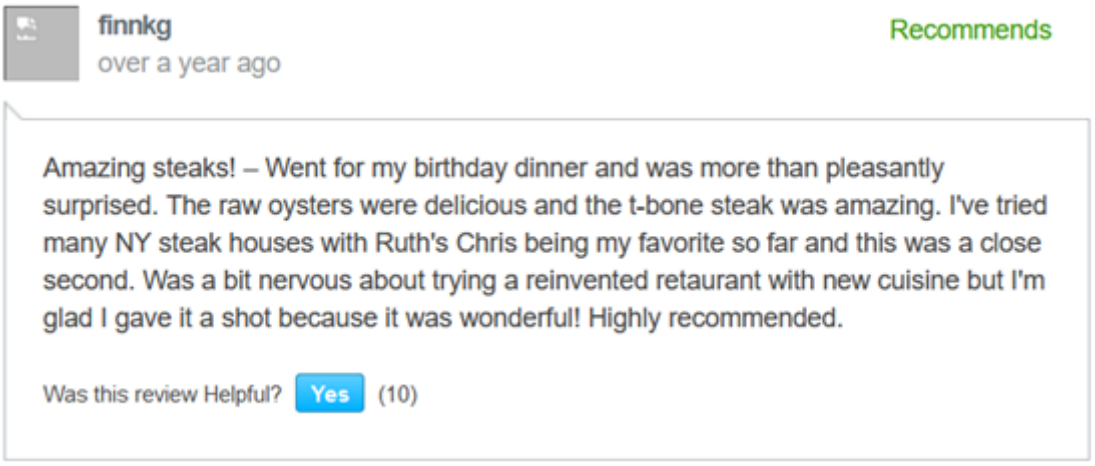


FIGURE 12 – Exemple de commentaire tiré de Citysearch
Source : http ://www.citysearch.com

Nous profitons de ces données étiquetées pour mieux mesurer les performances d'une solution. Les sources sont rassemblées dans le tableau 2 :

| Sources | Liens |
|---|---|
| **Citysearch** | http://spidr-ursa.rutgers.edu/datasets |
| **SemEval 2014** | https://alt.qcri.org/semeval2014/task4/index.php?id=data-and-tools |
| **SemEval 2015** | https://alt.qcri.org/semeval2015/task12/ |
| **SemEval 2016** | https://alt.qcri.org/semeval2016/task5/index.php?id=data-and-tools |
| **Yelp** | https://www.yelp.com/dataset/download |

TABLE 2 – Liens vers les sources de données

Ces données sont pré-labélisées en aspects et sentiments par aspect, à l'exception du jeu de données de Yelp. Ce dernier est utilisé comme ressource externe afin de fournir plus de commentaires textuels.

### 4.1.3 Séparation des jeux de données

Afin de simuler les nouvelles données qu'une solution n'a jamais vu, nous allons séparer les données en trois parties :

- **Un jeu de données d'entraînement**, constitué des commentaires du jeu du SemEval (de 2014 à 2016 confondues).

- **Un jeu de données de validation**, constitué des commentaires du jeu du SemEval (de 2014 à 2016 confondues).
- **Un jeu de données de test**, constitué des commentaires du jeu du SemEval (de 2014 à 2016 confondues) ainsi que le jeu de donnée extrait de Citysearch.

Ainsi, toute solution ou système sera entraîné sur le jeu de données d'entraînement. Les paramètres optimaux seront choisis pour maximiser les performances sur le jeu de validation. Pour une solution non-supervisée, il y aurait peu de sens de garder cette séparation. Dans ce cas, nous allons omettre le jeu de validation pour optimiser les paramètres directement depuis le jeu d'entraînement. Le jeu de test est à l'évaluation finale d'une solution.

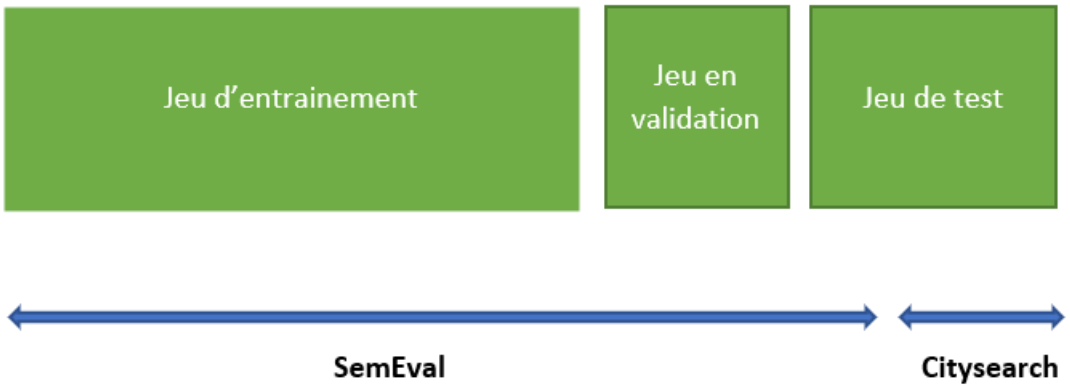


FIGURE 13 – Schéma synthétisant la segmentation des jeux de données

## 4.2 Matériel

Tout le code du projet a été réalisé en Python.

## 4.3 Mise en qualité du corpus

### 4.3.1 Segmentation des commentaires en phrases

Nous allons commencer par fragmenter les différents commentaires en phrases. La figure 14 donne un exemple. Nous supposons que la notion de phrase est suffisamment générale pour contenir au moins un aspect et un sentiment.

La segmentation en phrase peut devenir compliquée dès lors que le texte est mal construit (début de phrase pas en majuscule, pas de ponctuation finale, une variété de ponctuations

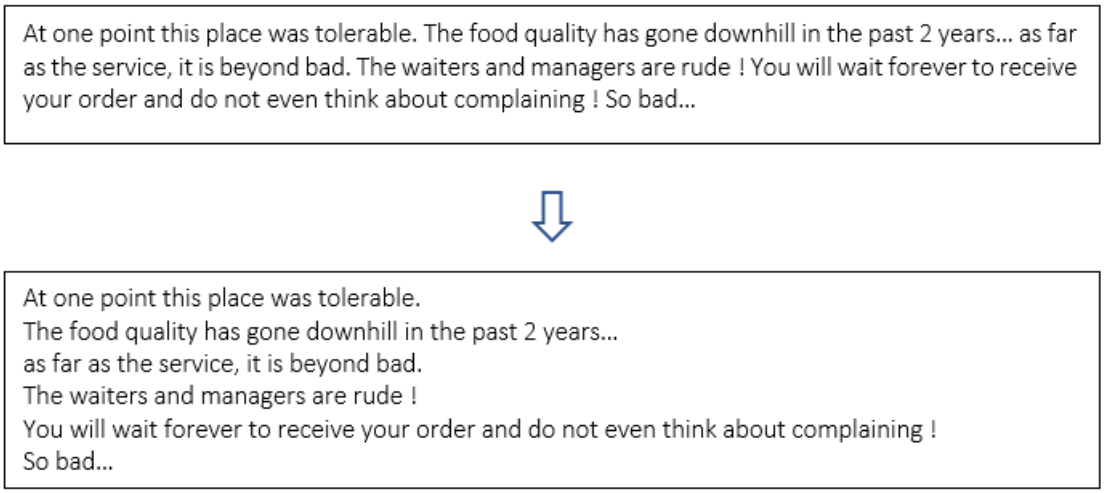


FIGURE 14 – Exemple de segmentation en phrases

finales, noms propres donnant lieu à des majuscules trompeuses, etc.) tel que l'exemple dans l'expérience qui suit.

À cet effet, plusieurs méthodes provenant des librairies Python ont été testées. L'expérience résumée dans la figure 15 montre la comparaison sur trois services. Le texte d'entrée devrait idéalement être séparé en trois phrases.

| **Service** | **Résultats** |
|---|---|
| NLTK | `Great coffee and pastries... Baristas are ?`<br>`and, E. Coli is a bacteria` |
| PySBD | `Great coffee and pastries...`<br>`Baristas are ?`<br>`and`<br>`E. Coli is a bacteria` |
| Spacy | `Great coffee and pastries...`<br>`Baristas are ?`<br>`and, E. Coli is a bacteria` |

FIGURE 15 – Résultat de la segmentation d'une phrase test avec différentes libraires spécialisées pour comparaison.

Finalement, Spacy est la solution qui répond le mieux à nos attentes.

### 4.3.2 Nettoyage du texte

Cette partie consiste à mettre le corpus en qualité afin de réduire le bruit parasite qui s'y cache pour affiner les analyses et les résultats de toute solution. Pour cela, nous cherchons à garder uniquement les termes pertinents pour l'analyse des sentiments et la recherche d'aspects (voir la figure 16). Ainsi, nous effectuons les étapes suivantes :

- Décontracter les expressions : « you're » en « you are » . . . ,
- Lemmatiser les termes, pour obtenir la forme canonique du mot,
- Retirer les caractères spéciaux, chiffres et ponctuations,
- Corriger les fautes d'orthographe,
- Supprimer les « stopwords ».

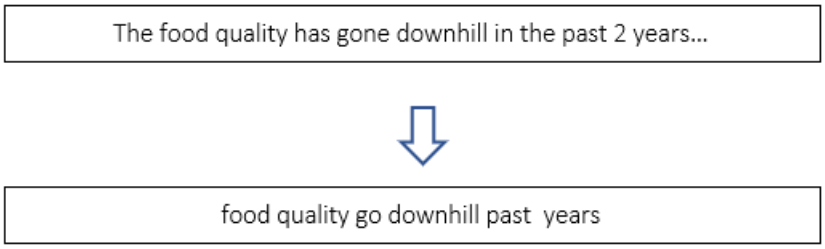


FIGURE 16 – Exemple de nettoyage de texte.

Les stopwords ont été sélectionnés à la main selon les termes qui apparaissaient dans le corpus d'entraînement. Ils ont été jugés non-pertinents pour estimer l'opinion tels que les déterminants ou encore les pronoms. Les termes visant à exprimer un contraste comme « but », « however » ont été gardés car ils peuvent aider à mieux estimer le sentiment.

### 4.3.3 Filtrage du texte

Le nettoyage des phrases a pu engendrer quelques « défauts » dans le corpus tels qu'une phrase devenue complètement vide (tous ses mots ont été retirés car supposés non-pertinents par exemple) ou ne contenant qu'un seul mot. Par ailleurs, le texte aurait pu être rédigé dans une langue différente que celle visée dans cette étude. De ce fait, nous pourrons utiliser le service proposé par Fasttext [2] pour l'identification de langue.

Pour cela, nous allons retirer les phrases possédant ces « défauts ».

2. https ://fasttext.cc/blog/2017/10/02/blog-post.html

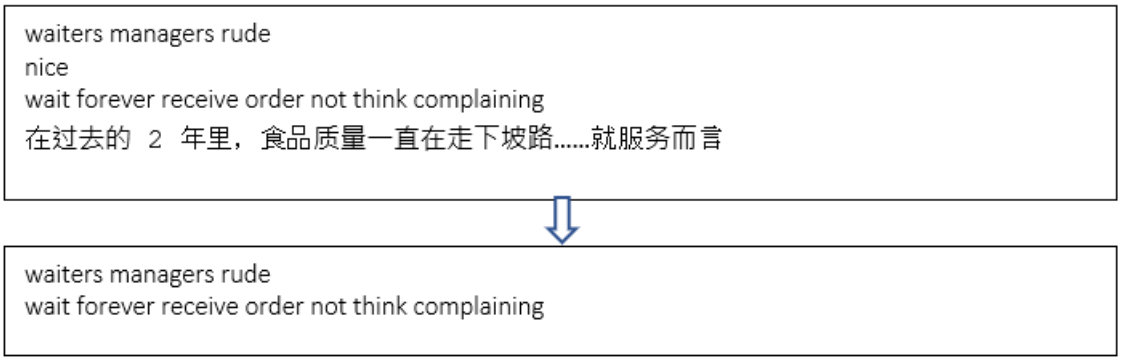


FIGURE 17 – Exemple de filtrage des textes.

Par ailleurs, les séquences longues contiennent très généralement peu d'information pertinente (souvent pour décrire le déroulé du repas). Ainsi, nous retirons les phrases contenant plus de 20 mots.

## 4.4 Analyse exploratoire des commentaires

Nous effectuons les analyses sur les données d'entraînement dans le but de pouvoir identifier des caractéristiques qui pourraient nous aider à résoudre la problématique.

Les phrases sont généralement très courtes, avec concentration entre 4 et 8 mots. La figure 18 montre la distribution du nombre de mots dans les différentes phrases.

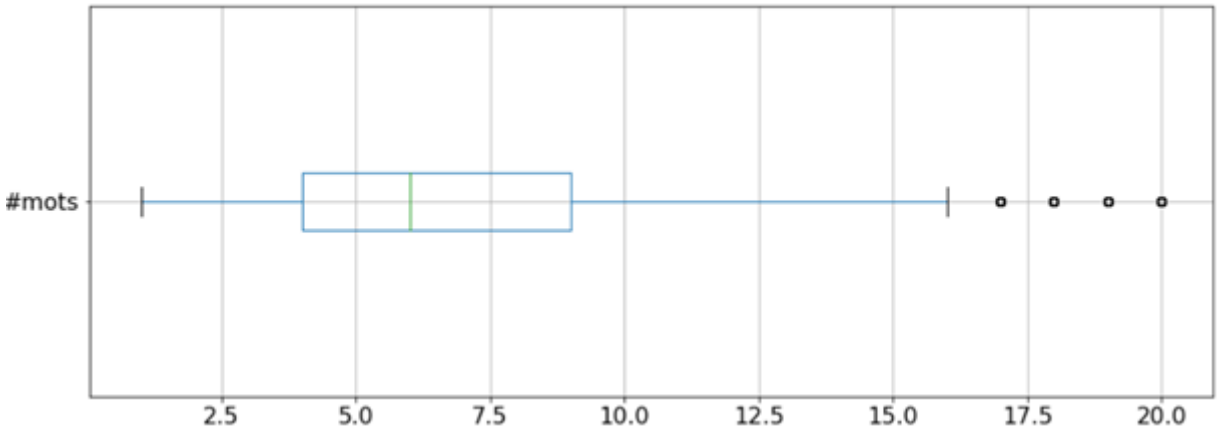


FIGURE 18 – Répartition du nombre de mots après nettoyage.

Sur la figure 19, nous observons que le corpus possède au moins 8 aspects fournis dont le plus prépondérant concerne des critiques sur la nourriture. La répartition des aspects n'est donc pas équilibrée. En outre, certaines phrases contiennent un sujet inclassable qui est représenté par l'aspect dénommé « anecdotes/miscellanous ».

Certains aspects sont assez distinguables : les termes les plus fréquents parmi leur sous-

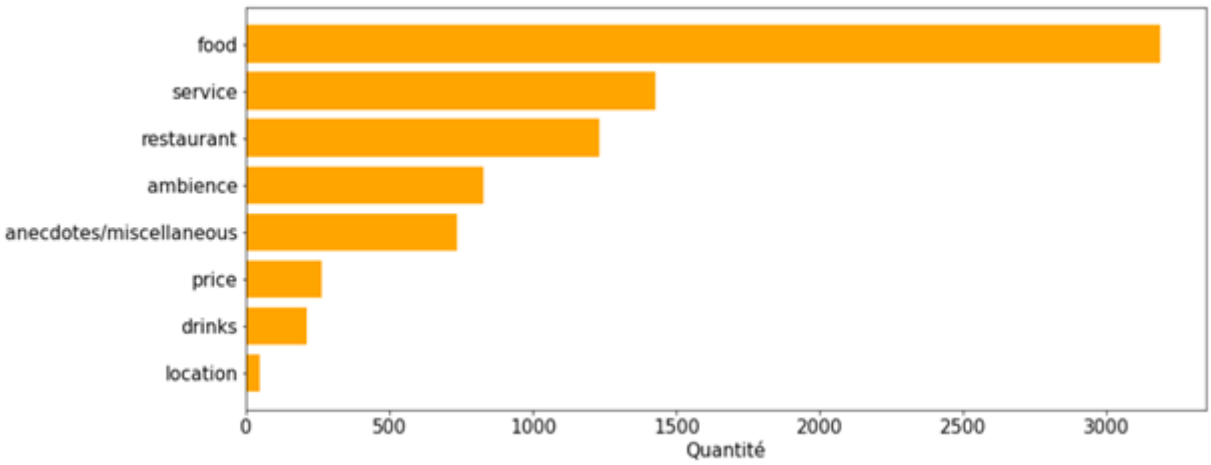


FIGURE 19 – Répartition des aspects prédéfinis.

corpus correspondants sont très spécifiques à la vue des nuages de mots (Figure 20). C'est le cas pour la nourriture, le service, voire l'ambiance. Par exemple, pour l'aspect « service », nous retrouvons les termes « service », « staff », « manager », « waitress », etc. En revanche, les aspects « miscellanous » et « restaurant » apparaissent relativement proches au regard des termes employés. De même, « price » et « drink » semblent très semblables.

Par ailleurs, nous remarquons que les noms communs sont les termes qui ressortent le plus au sein de chaque nuage de mots. Nous pouvons en déduire que ces derniers sont les principaux termes qui sont porteurs d'aspects.


FIGURE 20 – Nuages de mots par aspect

Au niveau de la distribution du nombre d'aspects, nous voyons sur la figure 21 que la plupart des phrases abordent un seul aspect. Cependant, près de 20% des phrases comportent au moins 2 aspects.

En observant la répartition du nombre de sentiments par phrase dans la figure 21, nous pouvons voir que la majorité renferme un seul sentiment (la phrase est soit positive, soit négative). Un faible pourcentage possède un double sentiment.

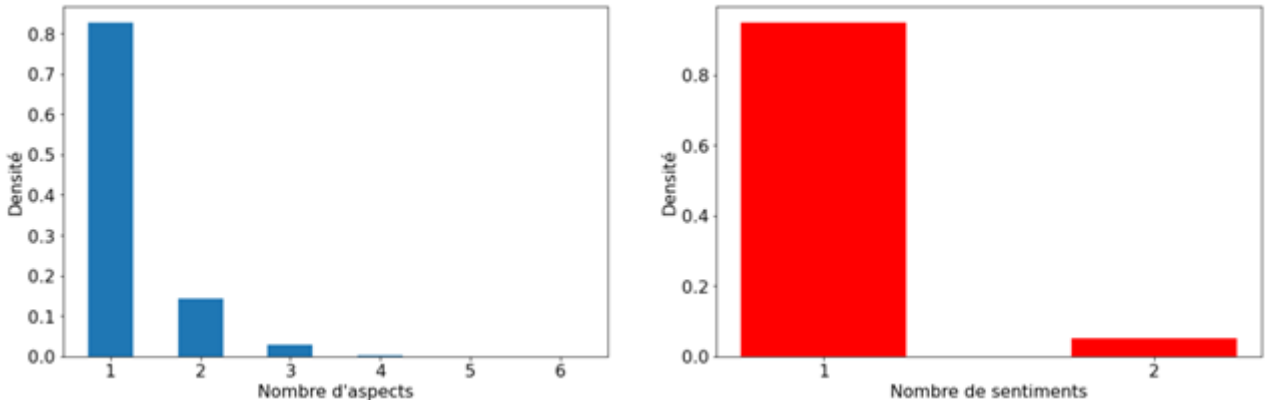


FIGURE 21 – Distribution du nombre d'aspects et du nombre de sentiments

Parmi les phrases contenant deux sentiments (voir figure 22), la majorité renferme également deux aspects. Cela laisse supposer qu'un aspect sera souvent associé à un sentiment.

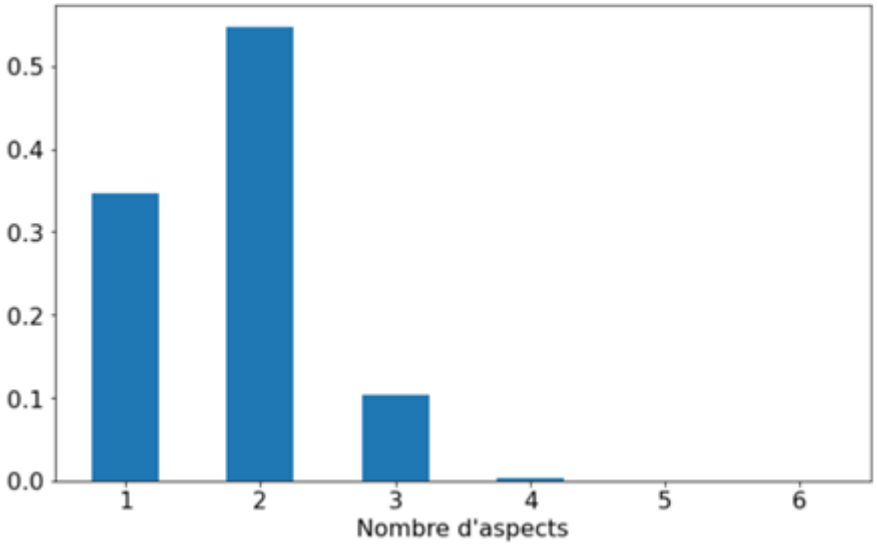


FIGURE 22 – Répartition du nombre d'aspects parmi les phrases à deux sentiments

La figure 23 un échantillon de phrases (sans nettoyage) possédant deux sentiments :

- Faan's got a great concept but a little rough on the delivery.
- The price is reasonable although the service is poor.
- This place is really trendi but they have forgotten about the most important part of a restaurant, the food.
- All the money went into the interior decoration, none of it went to the chefs.
- The rest of the dim sum is worth it though pricey by Chinatown standards.

FIGURE 23 – Échantillon de phrases possédant deux sentiments

Nous remarquons que le sentiment évolue du début de la phrase jusqu'à la fin : **les sentiments sont regroupés (par exemple, un début positif puis une fin négative)**. Néanmoins,

il est possible de distinguer ce changement qui a lieu régulièrement autour des **termes de contraste** tels que « but », « though », « however », « although », etc. Ainsi, il semblerait possible d'isoler le sentiment.

## 4.5 Hypothèses

Nous allons formuler les hypothèses suivantes :

- Les sentiments sont isolés au sein d'une phrase et peuvent être séparés par une coupure au milieu de la phrase.
- Un sentiment est toujours associé à un aspect ou plus dès lors que la phrase est suffisamment longue.
- Les phrases longues (comportant plus de 20 mots) ne sont pas pertinentes.
- Les phrases trop courtes (contenant qu'un seul mot) ne peuvent pas contenir à la fois un aspect et un sentiment.

## 4.6 Introduction de l'approche

### 4.6.1 Explication du système général

Afin de pouvoir repérer un aspect et visualiser le sentiment par aspect, nous proposons une approche reposant sur deux étapes après nettoyage du corpus.

- La première étape consiste à **isoler les sentiments dans des segments de phrases**. Par hypothèse, une phrase peut contenir deux sentiments séparables par une coupure. Le but de cette étape est donc de diviser la phrase, si nécessaire, afin d'avoir un sentiment par segment.
- La seconde étape consiste à **retrouver les aspects évoqués au sein de chaque fragment de phrases** engendré précédemment.

Le système est résumé sur la figure 24

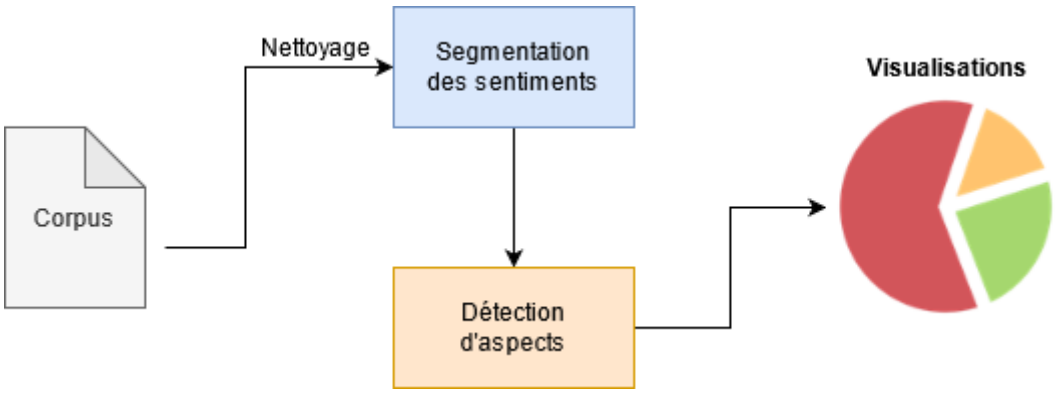


FIGURE 24 – Schéma de l'approche

| Caractéristiques | Valeurs |
|---|---|
| **Nombre de phrases** | 2 198 632 |
| **Taille du vocabulaire (nombre de mots uniques)** | 109 905 |

TABLE 3 – Caractéristiques des données d'entraînement du Word2Vec

### 4.6.2 Modèle de représentations vectorielles utilisé

TULKENS et CRANENBURGH 2020 ont démontré l'importance de choisir un modèle d'embedding entraîné sur le même domaine que la tâche visée. En effet, ils ont illustré une chute importante dans la performance de leur algorithme dès lors qu'un modèle plus généraliste était employé.

De ce fait, nous allons mettre à notre disposition un modèle de représentation vectorielle adapté au domaine sur lequel nous travaillons. Pour cela, un modèle Word2Vec est choisi et entraîné sur des documents issus du domaine de la restauration.

Pour former le jeu de donnée d'entraînement, nous allons récupérer les données du jeu de Yelp et du SemEval (partie entraînement). Cela permet d'obtenir un grand volume de données dont les caractéristiques sont résumées dans le tableau 3

Les paramètres d'entraînement du modèle sont donnés dans le tableau 4. La librairie Gensim a été utilisée pour entraîner ce modèle.

La figure 25 montre une représentation vectorielle d'un sous-ensemble de mots initia-

| Paramètres | Valeurs |
|---|---|
| **Architecture** | CBOW |
| **Taille de la fenêtre (window)** | 7 |
| **Taille des vecteurs** | 200 |
| **Nombre d'epochs** | 10 |
| **Répétition minimale d'un mot** | 2 |

TABLE 4 – Paramètres d'entraînement du modèle Word2Vec

lement répartis en 4 catégories (prix, nourriture, service, ambiance). La dimension de leur représentation vectorielle a été réduit en utilisant l'algorithme UMAP (McInnes, Healy et Melville 2020) afin de pouvoir afficher leurs coordonnées sous deux dimensions. Il est possible d'observer que les termes ayant un sens proche ont une position voisine dans l'espace de projection. Par exemple, les mots « sushi », « tapas » et « food » se trouvent en centre du cluster catégorisant la nourriture. La couleur indique la catégorie d'appartenance réalisée en amont.

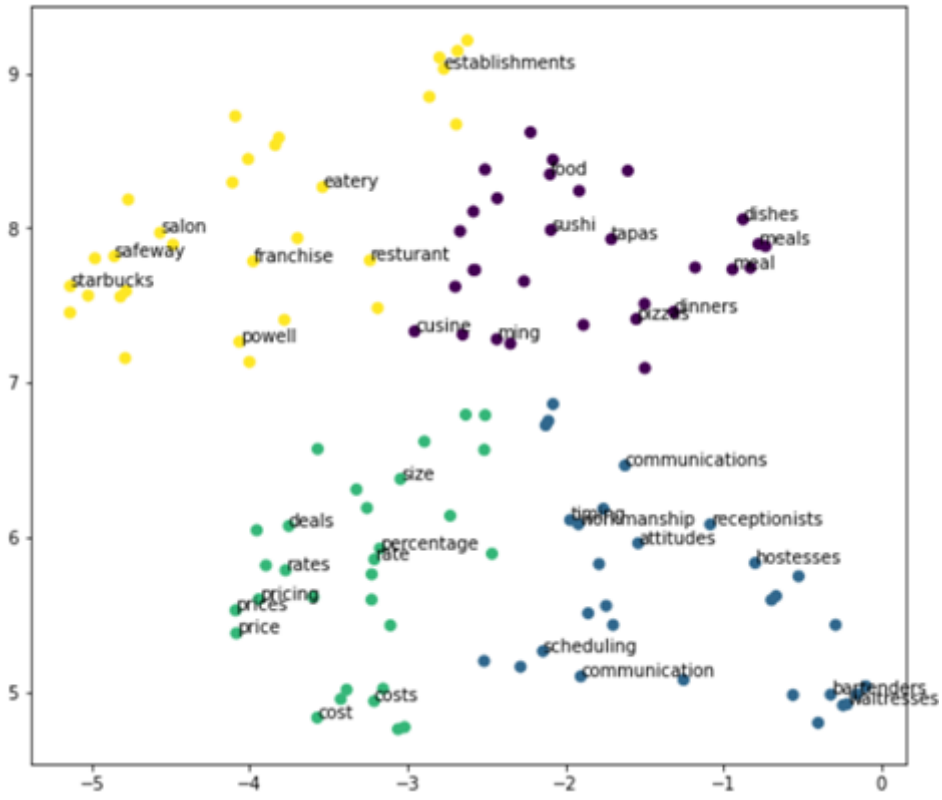


Figure 25 – Représentation vectorielle des mots projetés en deux dimensions.

Le tableau 5 montre le score de similarité entre des paires de mots. La représentation vectorielle est donnée par le modèle Word2Vec entraîné dans cette section. Nous pouvons remarquer que les termes ayant une sémantique proche ont tendance à avoir un score de similarité proche de 1. Cela signifie que ces vecteurs sont proches dans l'espace d'encodage. À l'inverse, les paires de termes possédant des sémantiques différentes (comme "pizza" et "ambience") ont un score de similarité faible (proche de zéro).

| Paires de mots | Similarité (cosinus) |
|---|---|
| **staff-service** | 0.51 |
| **food-pizza** | 0.48 |
| **pizza - ambience** | 0.03 |
| **waiter - waitress** | 0.87 |

TABLE 5 – Similarités entre des paires de mots

## 4.7 L'analyse des sentiments

### 4.7.1 Modélisation du problème

Le but de cette partie est de pouvoir séparer une phrase pour avoir un sentiment par fragment. À cet effet, nous allons supposer qu'un mot sera toujours associé à une polarisation positive ou négative. Pour cela, nous allons **modéliser le problème comme une classification de tokens**. Le but est de prédire une séquence de tokens en sortie à partir d'une séquence en entrée qui auront la même taille. La séquence d'entrée est la phrase à traiter, découpée en token qui représentera les mots d'un document. La séquence de sortie détermine la polarité des termes de la séquence d'entrée. Le schéma dans la figure 26 en illustre un exemple.

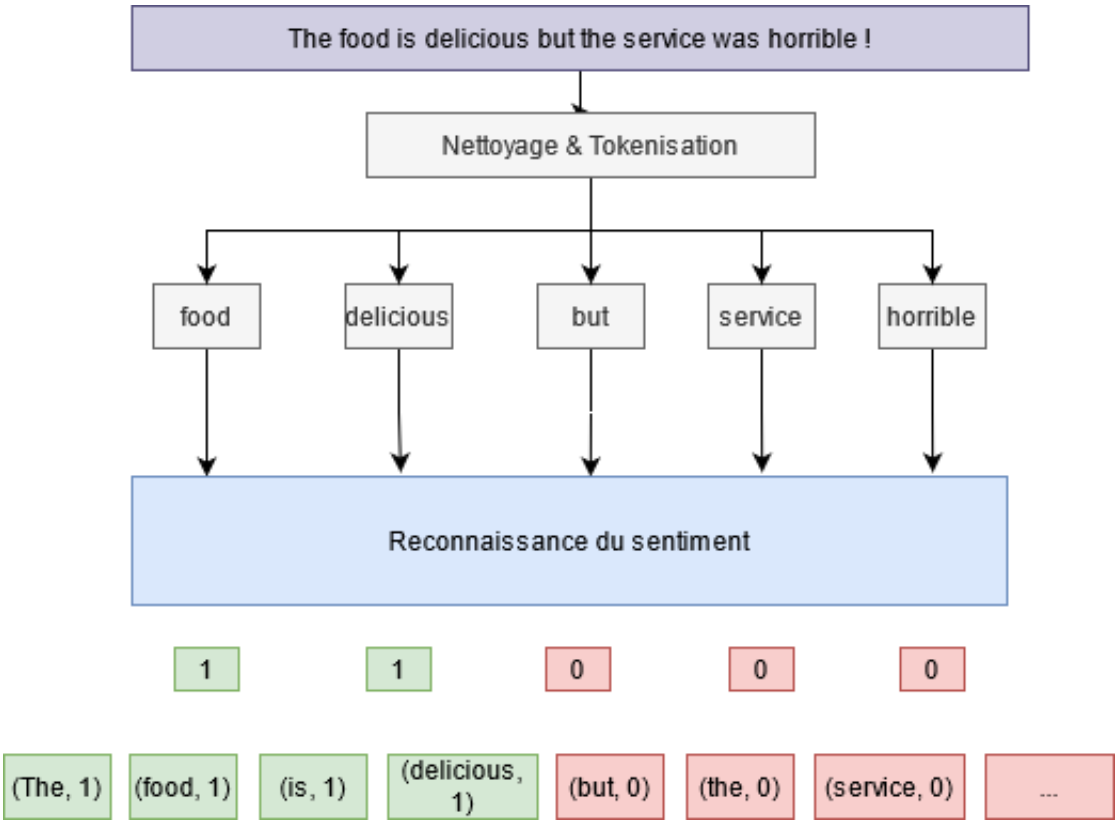


FIGURE 26 – Schéma représentant la segmentation d'une phrase.

Après nettoyage du texte, ce dernier est fragmenté en token. Chaque token pourra être

encodé en son embedding puis traité par un modèle de reconnaissance de sentiments pour prédire si les termes sont positifs ou négatifs. Nous noterons par :

- un label 1 le sentiment positif d'un terme,
- un label 0 le sentiment négatif d'un terme.

Puisque certains ont été filtrés par le processus de nettoyage, la dernière étape consiste à interpoler les sentiments aux les termes qui ont été retirés.

### 4.7.2 Métrique utilisée

Afin d'évaluer un modèle de segmentation sur cette tâche, nous allons nous baser sur **le taux d'erreur**, jugé suffisamment fiable étant donné que les classes ont été rééquilibrées.

$$erreur = 1 - \frac{\#bonnes\ reponses}{nombre\ d'exemples} \tag{16}$$

### 4.7.3 Génération de données

Pour la prédiction de séquence de sentiments, aucun jeu de données externe n'a pu être trouvé. De ce fait, nous allons construire nos propres données d'entraînement à partir des données de Yelp et du SemEval. Le but est d'obtenir des séquences de tokens avec leur sentiment correspondant. La figure 27 illustre l'exemple d'une séquence de tokens voulue.

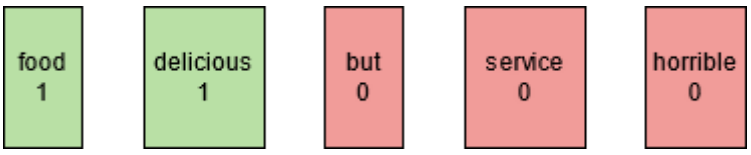


FIGURE 27 – Exemple de séquence d'entrée pour l'apprentissage de la segmentation. (vert : positif, rouge : négatif)

Pour débuter, nous allons reprendre les données du SemEval et ne garder que les phrases comportant un seul sentiment. Par la suite, nous allons traiter les données de Yelp labélisées. Elles sont annotées avec un nombre d'étoiles compris entre 1 (très négatif) et 5 (très positif) attribué par le rédacteur du commentaire. À partir de ces commentaires, nous allons les segmenter en phrases. Le label de la phrase est le même que le label du commentaire auquel il est rattaché. La figure 28 montre un échantillon de commentaires et le nombre d'étoiles associé.

| Commentaire | Nombre d'étoiles |
|---|---|
| A disappointment. The food was lackluster at best and the service was terribly slow. I wouldn't go back even with a Groupon. | 1 |
| Okay place, Terrible wait time. Good atmosphere, decent food. I don't understand a Saturday brunch menu until 3pm. I won't be back | 2 |
| Cute place. Slow service but really good food with friendly staff! | 3 |
| Best place for wings and beer! Service is always great and food is always fresh. My 5 year old daughter loves the chicken tenders and frys's. | 4 |
| Perfect spices, nice atmosphere & music, great lighting and decor. Even the restroom area is pretty cool. Will definitely come again. | 5 |

FIGURE 28 – Échantillon de commentaires de Yelp classés selon leur note.
Les couleurs distinguent le sentiment

Nous voyons que les commentaires ayant une note de 2 ou 3 comportent des phrases positives et négatives. Ce sont des notes pour attribuer un avis mitigé. Afin d'éviter un maximum d'erreur de label, nous allons uniquement retenir les commentaires ayant une note de 1 ou 5 et nous pouvons supposer que chacune de leurs phrases partagent le même sentiment. Par ailleurs, nous retirons les phrases présentant une longueur excessive. Ainsi, nous recueillons près de 75000 phrases provenant de Yelp et près de 3000 phrases du SemEval. La distribution des labels au sein du jeu de données formé est équilibrée. La figure 29 résume la situation.

Par la suite, nous souhaitons obtenir des phrases ayant une double polarité. Pour cela, nous allons concaténer des phrases choisies aléatoirement. D'un point de vue programmatique, il est possible de créer un générateur [3] ayant pour but de **choisir deux phrases, quel que soit leur sentiment, et de les fusionner pour donner l'illusion d'avoir une seule phrase.** C'est la raison pour laquelle les phrases longues ont été retirées afin d'obtenir du texte avec une taille raisonnable. De plus, le choix d'un générateur a l'avantage d'engendrer des nouvelles paires. Le schéma sur la figure 30 résume la manière dont les données sont générées.

3. Générateur avec Python et Keras : exemple

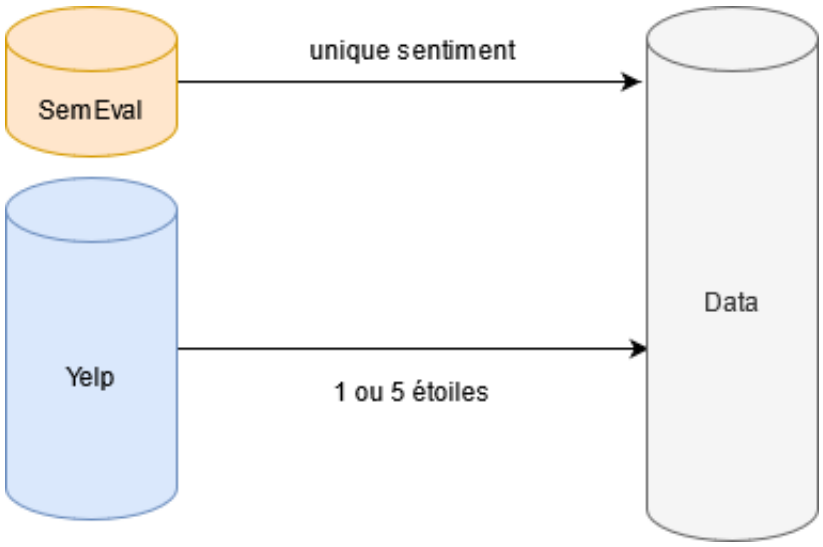


FIGURE 29 – Provenances des données pour la segmentation et filtres appliqués.

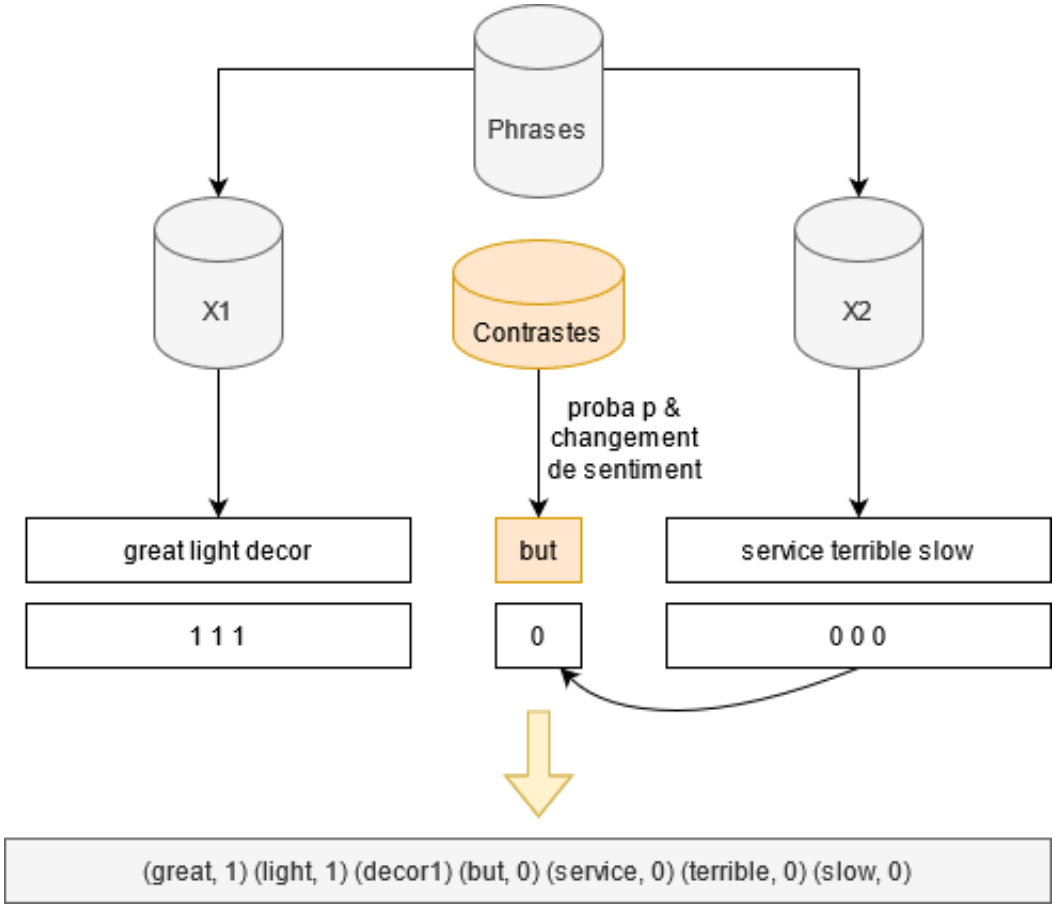


FIGURE 30 – Schéma de la génération de séquences de termes labélisés en sentiment.

Notons les étapes de la génération :

- Les phrases sont partitionnées aléatoirement en deux sous-ensembles $X_1$, $X_2$. Chacun de ces sous-ensembles contient autant de phrases positives que de phrases négatives.
- Un ensemble de termes de contrastes est créé et contient les termes suivants : but, however, although, though, despite, while.

- Pour la génération, une phrase est tirée aléatoirement de $X_1$ et une autre de $X_2$ dont leur label est connu. S'il y a un changement de sentiment (par exemple, la première phrase est positive et la seconde négative), **un terme de contraste est tiré avec une probabilité p**, fixée à 0.5. Choisir une valeur plus grande entraîne un sur-apprentissage où le modèle se base uniquement sur le terme de contraste pour changer le sentiment.
- Si un terme de contraste est tiré, son label est désigné comme étant le même que la seconde phrase.
- Les deux unités textuelles, en ajoutant éventuellement le terme de contraste, sont concaténées pour former la séquence d'entrée.

De cette manière, une phrase générée peut être soit :

– entièrement positive,

– entièrement négative,

– postive puis négative,

– négative puis positive.

### 4.7.4 Modèle et résultats

Pour répondre au problème de classification de tokens, nous allons entraîner un modèle d'apprentissage profond en utilisant des couches BiLSTM. Un réseau LSTM (Long Short-Term Memory) est un réseau de neurones récurrent, créé pour répondre aux problèmes de propagation du gradient. Il encode un token de la séquence en se basant sur les tokens qui le précèdent. Cela permet d'ajouter le contexte donné par les tokens passés.

Le BiLSTM (pour Bidirectional LSTM) est une concaténation de deux LSTM. Le premier lit la séquence dans le sens courant, l'autre dans le sens inverse. Cela donne lieu à deux interprétations de la même séquence qui seront concaténées terme à terme pour former un seul encodage. Ainsi, la représentation d'un token de la séquence prend en compte le contexte passé et futur. Le mécanisme du BiLSTM est récapitulé dans la figure 31.

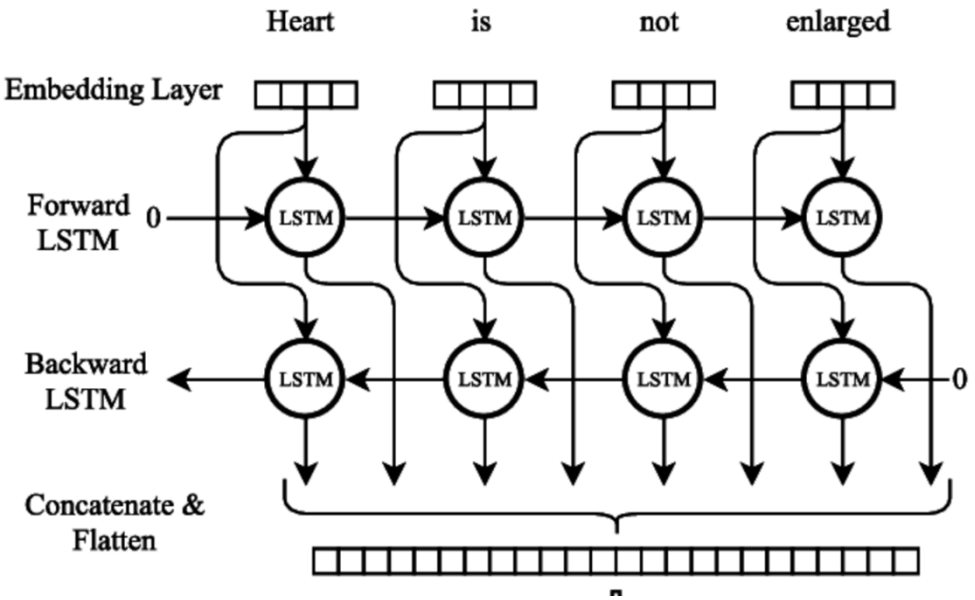


FIGURE 31 – Illustration d'un BiLSTM.
Source : https ://paperswithcode.com/method/bilstm

| Hyperparamètres | Valeurs |
|---|---|
| **Taille des batchs** | 128 |
| **Pas d'apprentissage** | 0.0005 |
| **Algorithme d'optimisation** | Adam |
| **Fonction de coût** | Binary crossentropy |
| **Régularisation** | L2 |

TABLE 6 – Hyperparamètres de l'entraînement du réseau BiLSTM

Dans notre cas, l'idée d'ajouter l'information du contexte passé et futur est intéressant dans la mesure où le sentiment d'un terme peut autant dépendre de la suite de la séquence que du contexte passé. Dans la situation d'un changement de sentiment, l'information du contexte futur apparaît utile. C'est donc ce qui motive le choix d'un modèle basé sur un BiLSTM.

Pour encoder un terme d'une séquence donnée, nous allons employer la représentation vectorielle du Word2Vec entraîné précédemment. Les hyperparamètres choisis pour l'entraînement sont résumés dans le tableau 6.

L'architecture du réseau de neurones est décrite dans la figure 32. Les détails sont laissés en annexes.

Pour réduire le phénomène de sur-apprentissage, chaque couche BiLSTM est associée à une régularisation afin de pénaliser le modèle dès lors que ses paramètres s'éloignent de 0. Les couches de Dropout ont été introduites dans la même optique. Des améliorations dans l'évolution des performances ont été constatées et l'entraînement est devenu plus stable

(moins d'oscillations dans la progression des coûts). De plus, le EarlyStopping a été utilisé afin d'arrêter l'entraînement pour revenir aux paramètres ayant généré les meilleurs scores (après 3 epochs sans progression).

Notons que la mise à jour des paramètres ne s'effectue pas dans la matrice d'embedding où les poids sont fixés. Le modèle a été mis en place avec les librairies Keras et Tensorflow. Les figures 33 et 34 retracent l'évolution des performances durant l'entraînement.

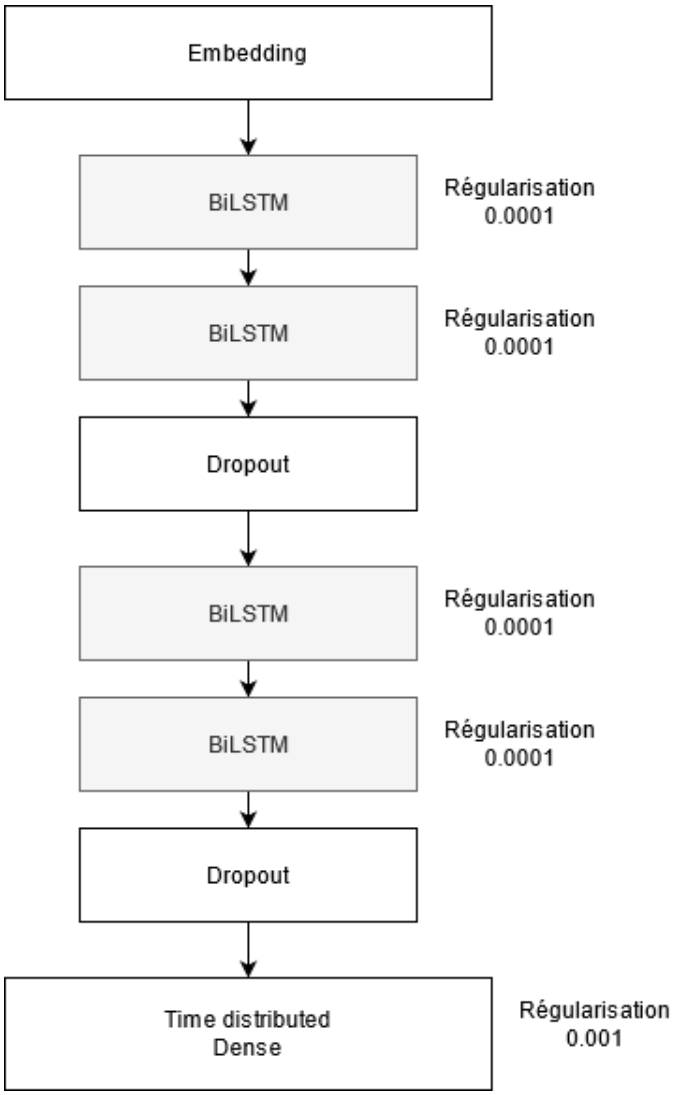


FIGURE 32 – Architecture du réseau BiLSTM utilisée

L'entraînement s'est déroulé sur 15 epochs. Le modèle a un taux d'erreur final de 15,46%. Par ailleurs, on voit que la courbe traçant les performances sur les données de validation se sépare de celle des données d'entraînement au fur et à mesure que les epochs progressent. Cela témoigne d'une difficulté à généraliser et une possible limitation du modèle utilisée.

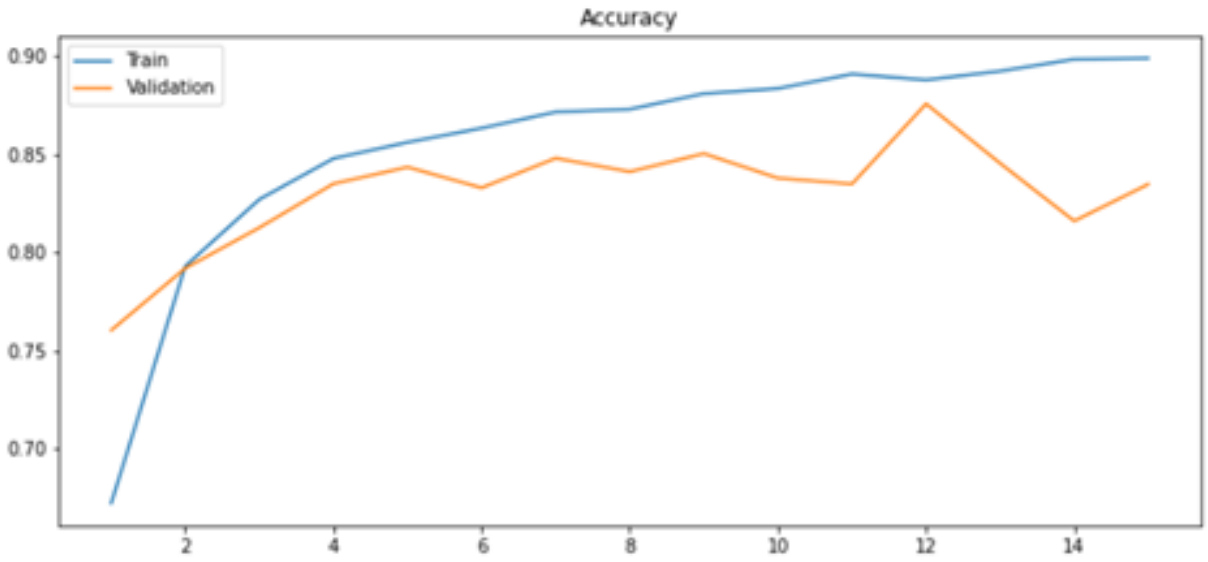


FIGURE 33 – Evolution du taux de bonnes réponses du modèle BiLSTM
La courbe bleue représente l'évolution sur les données d'entraînement tandis que le courbe orange montre l'évolution sur les données de validation.

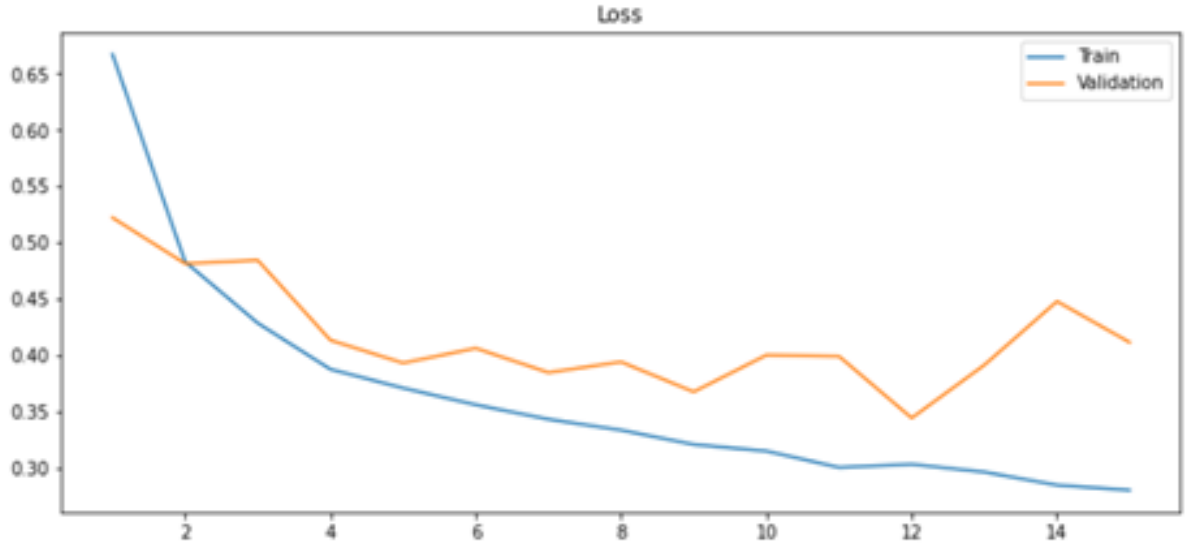


FIGURE 34 – Evolution des coûts du modèle BiLSTM
La courbe bleue représente l'évolution sur les données d'entraînement tandis que le courbe orange montre l'évolution sur les données de validation.

### 4.7.5 Interpolation de la polarité

Le modèle prédit le sentiment pour les termes qui ont été passés en entrée. Or, le nettoyage du texte filtre certains mots. Il sera utile pour la suite d'avoir une polarité pour l'ensemble des mots de la phrase d'origine (avant nettoyage). C'est la raison pour laquelle nous cherchons à interpoler la polarité dans le texte. L'exemple sur la figure 35 illustre le problème.

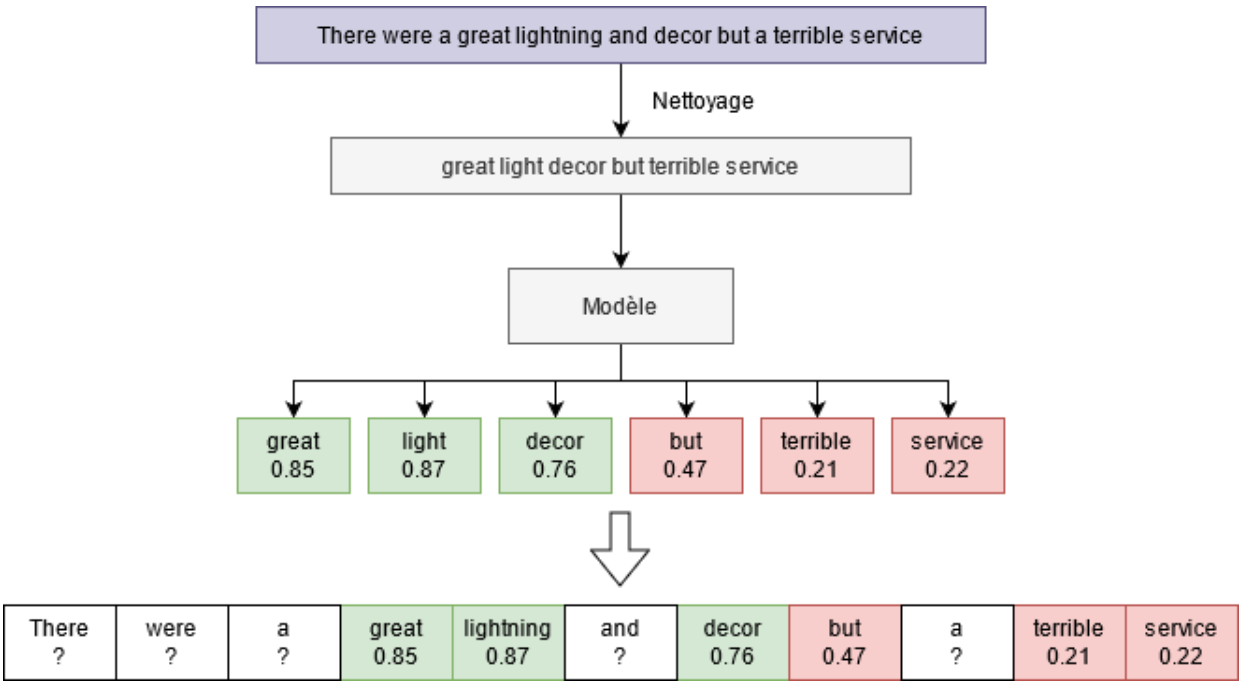


FIGURE 35 – Instance du problème d'interpolation

Pour compléter l'estimation du sentiment des termes, nous allons employer le concept de **glissement de fenêtre (sliding window).** Chaque terme dépourvu de sentiment aura une polarité estimée en observant la polarité des termes qui tombent dans sa fenêtre glissante. Par exemple, pour estimer le sentiment du mot « and » dans la figure qui suit avec une fenêtre de taille 1, nous allons considérer les termes qui sont à un pas de 1 de « and ». Cela correspond à la partie grisée sur la figure 36 :

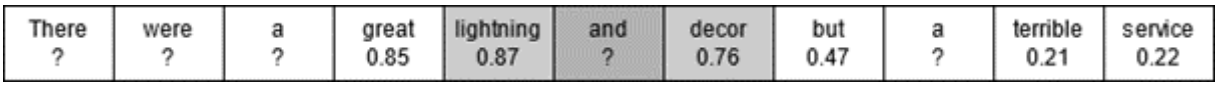


FIGURE 36 – Exemple d'interpolation sur un seul terme

La polarité du mot est déterminée par la moyenne des polarités voisines. La séquence suivante obtenue est illustrée sur la figure 37 :

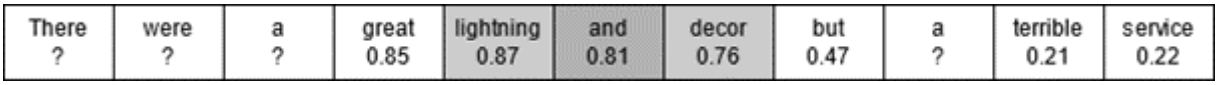


FIGURE 37 – Exemple d'interpolation sur le terme "and"

Cependant, dans le cas où la fenêtre glissante est vide (comme pour le terme « were » dans l'exemple précédent), aucune estimation n'est réalisée. Lorsque la fenêtre glissante aura atteint la fin de la phrase mais que des termes ne possèdent toujours pas de polarité, une nouvelle itération sera effectuée. Le processus continue tant que tous les termes n'ont pas de polarité. Le déroulé de l'interpolation est illustré dans la figure 38.

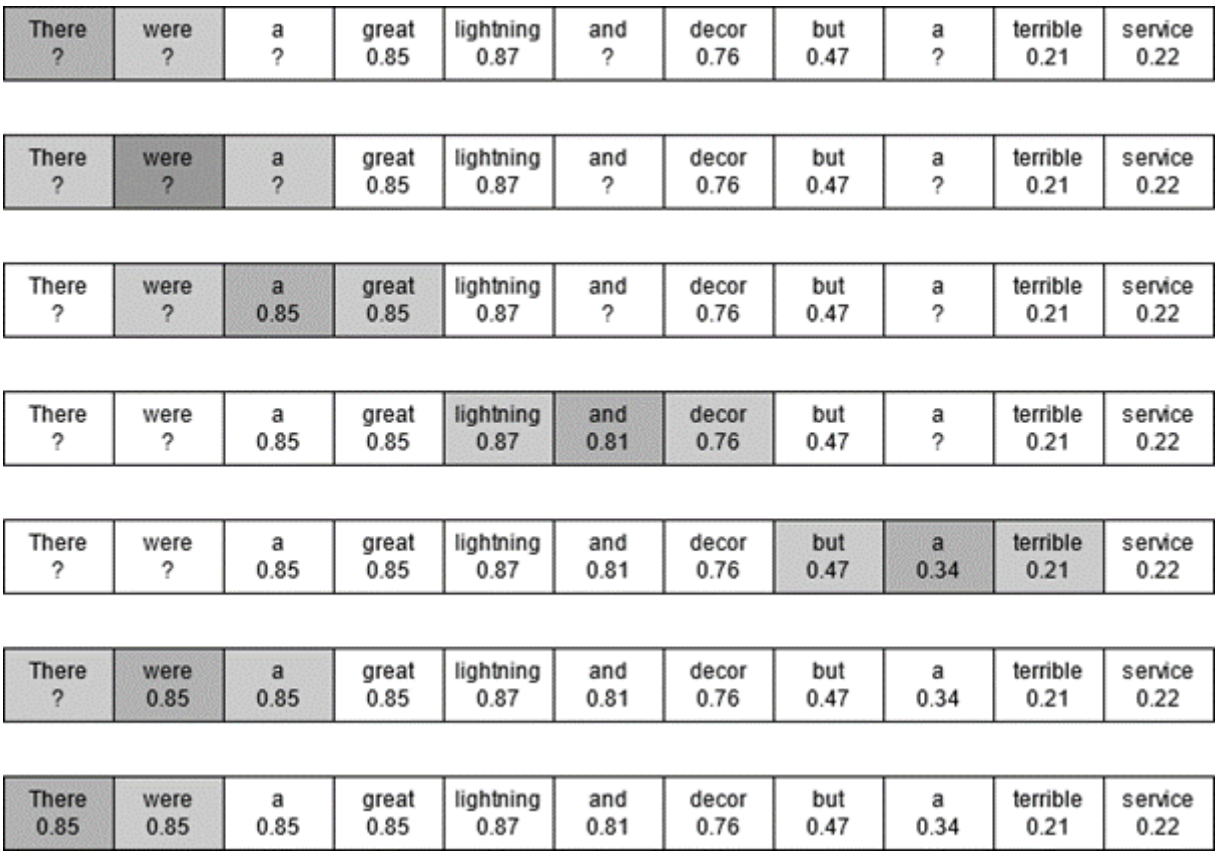


FIGURE 38 – Le déroulé de l'algorithme d'interpolation

Finalement, il est possible de segmenter la phrase selon les deux sentiments qui la composent. Le résultat est donné sur la figure 39 :


FIGURE 39 – Résultat final de la segmentation en sentiments

La phrase a été segmentée en une unité de texte contenant un seul sentiment et nous disposons du sentiment de chaque unité.

### 4.7.6 Analyse des erreurs et résultats

La figure 40 présente un échantillon des données de test coloriées selon le sentiment prédit par le modèle. La couleur verte indique un sentiment positif et la couleur rouge un sentiment négatif. Nous observons que le modèle a su séparer des phrases ayant un double sentiment. Par exemple, la segmentation a été parfaite sur les phrases n°1 et n°6. Par ailleurs, sur des phrases ayant un seul sentiment, le modèle pu prédire le même sentiment sur chaque token. À titre d'exemple, il est possible de regarder les phrases n°9 et n°10.

| | |
|---|---|
| 0 | the bad though be the taste |
| 1 | cool atmosphere but such a let down |
| 2 | i have to say they have one of the fast delivery time in the city |
| 3 | i do it know what some people who rave about this hot dog be talk about |
| 4 | we enjoy ourselves thoroughly and will be go back for the dessert |
| 5 | stay away if you be claustrophobic |
| 6 | great food but the service be dreadful |
| 7 | i complain to the waiter and then to the manager but the intensity of rudeness from they just go up |
| 8 | not terrible but not the restaurant in the review of do |
| 9 | as we wait i watch do separate group of diner discuss how disappointed they also be |
| 10 | have always have a great time here |
| 11 | the two waitress be look like they have be suck on lemon |
| 12 | i be starve and the small portion be drive i crazy |

FIGURE 40 – Prédiction des sentiments sur les données de test

En revanche, toutes les prédictions ne sont pas toujours correctes. La figure 41 affiche des phrases portant au moins une erreur commise par le modèle. Plusieurs types d'erreur se distinguent.

- Nous pouvons remarquer un terme de contraste en début de phrase : « while ».
- Certains termes ont été supprimé car ils étaient vu comme des stopwords tels que « even » et « if ». Ils prennent un sens important dans certains cas comme dans l'exemple « never disappointing even if the price be a bit over the top ».
- Certains mots n'ont pas été pris dans leur contexte. Dans la phrase précédente, « top » est positif contrairement au reste de la phrase.

Ces points soulignent un nettoyage du texte trop strict et une difficulté à comprendre le contexte. En effet, la phrase « i absolutely suggest this place » a été comprise négativement. Étrangement, le terme « suggest » à une connotation négative pour le modèle. Cela peut être un signe de sur-apprentissage où ce mot apparaît fréquemment parmi les phrases négatives.

La matrice de confusion ci-dessous indique que le modèle BiLSTM a une légère tendance à considérer des phrases négatives comme étant positive. Néanmoins, ses prédictions restent globalement justes avec **un taux de bonnes réponses de 85.45% après le processus d'interpolation**. Ces performances sont bien meilleures qu'un modèle aléatoire produisant

| | errors |
|---|---|
| 0 | while it be large and a bit noisy the drink be fantastic and the food be superb |
| 1 | while there be a decent menu it should not take ten minute to get your drink and do for a dessert pizza |
| 2 | the management be less than accommodate |
| 3 | all in all i would return as it be a beautiful restaurant but i hope the staff pay more attention to the little detail in the future |
| 4 | straight forward no surprise very decent japanese food |
| 5 | the food be excellent as well as service however i leave the four seasons very disappointed |
| 6 | decor need to be upgrade but the food be amazing |
| 7 | the dessert we have a pear torte be good but once again the staff be unable to provide appropriate drink suggestion |
| 8 | i know real indian food and this be not it |
| 9 | a mix of student and area resident crowd into this narrow barely there space for its quick tasty treat at dirt cheap price |
| 10 | try the rose roll not on menu |
| 11 | once we sail the top notch food and live entertainment sell we on a unforgettable evening |
| 12 | the fajita we try be tasteless and burn and the mole sauce be way too sweet |

FIGURE 41 – Échantillon d'erreur sur les sentiments

un taux d'erreur de 50%. Cela nous suffit pour valider ce modèle.

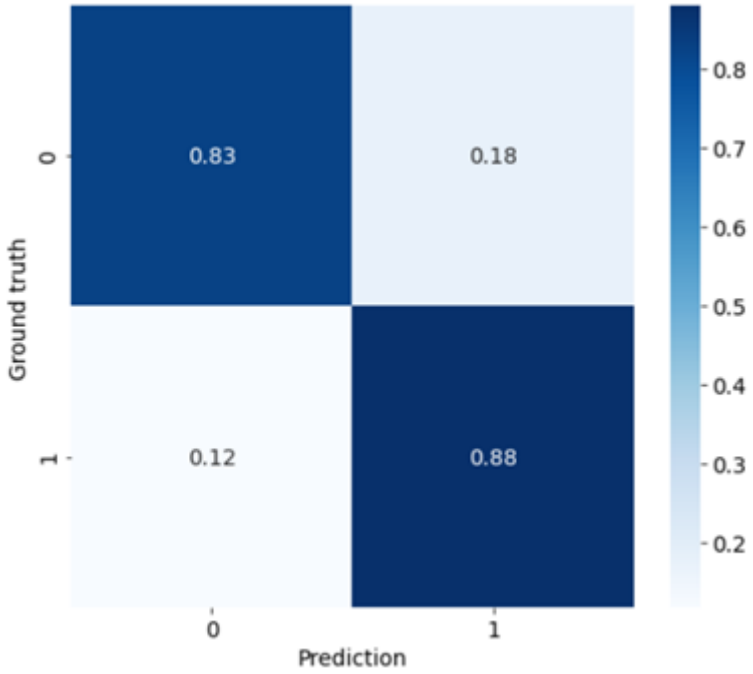


FIGURE 42 – Matrice de confusion du modèle BiLSTM sur les sentiments

## 4.8 Détection des aspects avec des mesures de similarité

### 4.8.1 Objectifs

La segmentation des commentaires en phrases puis en sentiments a permis d'isoler l'opinion dans une phrase. Ces fragments de texte peuvent renfermer plusieurs aspects. L'objectif de cette partie est d'identifier les aspects abordés au sein des différents segments de textes.

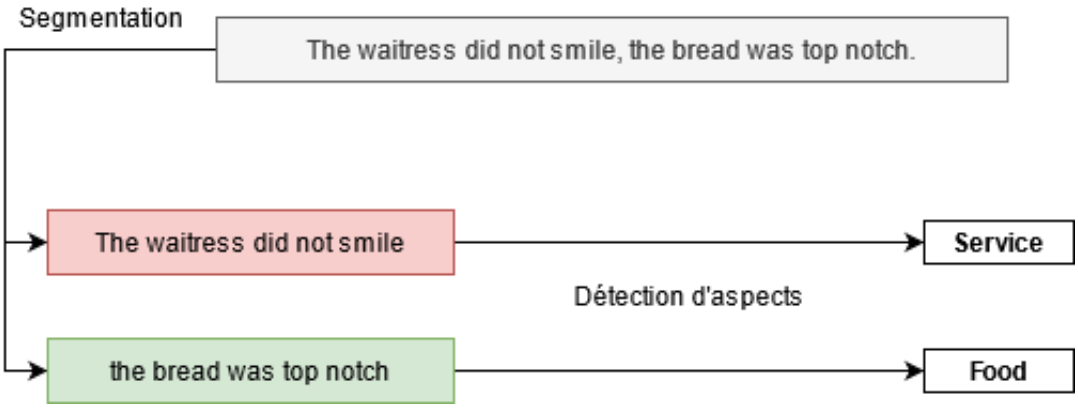


FIGURE 43 – Exemple de détection d'aspects sur une phrase avec deux sentiments

Afin de se donner les moyens d'y parvenir, **la liste des aspects est prédéterminée, comme un paramètre du système**. Les aspects sont considérés comme des paramètres de l'algorithme. Des exemples d'aspects peuvent être le service, la nourriture/les plats, le prix, l'ambiance, etc. Ils dépendent notamment du domaine et de l'intérêt de l'utilisateur du système. C'est la raison pour laquelle cela est laissé en tant que paramètre. **Ici, nous allons nous restreindre à un sous-ensemble d'aspects figurant dans nos données annotées : la nourriture, le service et l'ambiance.**

### 4.8.2 Méthodologie

Afin de se donner des moyens d'évaluation, les données du SemEval ainsi que de Citysearch sont utilisées. Il sera question de reprendre les données d'origine en gardant que les phrases ayant un seul sentiment. Cela permet d'avoir des données qui se rapprochent des segments de textes provenant de la séparation des sentiments.

Les données étant labélisées en aspects, il peut être judicieux de réutiliser ces étiquettes pour savoir si le modèle a réussi à associer le bon aspect à la bonne phrase. La labélisation de ces données n'étant pas équilibrée, certains aspects sont associés à plus de phrases que d'autres. Par exemple, l'aspect « food » est associé a plus de 3000 phrases tandis que

l'aspect « service » ne dépasse pas les 1500 phrases dans le corpus d'entraînement. Ce phénomène peut devenir gênant dans la mesure où un taux d'erreur ne suffit pas pour faire la différence entre les bonnes réponses du modèle et le manque de cas pour souligner les défauts du modèle. Pour faire en sorte que le déséquilibre des classes n'importune pas l'évaluation, **la métrique utilisée est le F1-score** définie dans la formule 19 en notant $TP$ : vrai positif, $TN$ : vrai négatif, $FP$ : faux positif, $FN$ : faux négatif. On gardera une grande importance sur **la précision** car on souhaite que la détection soit pertinente (les prédictions positives sont bien correctes).

$$precision = \frac{TP}{TP + FP} \tag{17}$$

$$recall = \frac{TP}{TP + FN} \tag{18}$$

$$f1 = \frac{2precision.recall}{precision + recall} \tag{19}$$

### 4.8.3 Adapter un algorithme existant

Plusieurs méthodes non-supervisées ont démontré leur efficacité en se basant sur des mesures de similarités entre les aspects choisis et les documents textuels. Une de ces méthodes qui s'est avérée efficace a été développée par Tulkens et Cranenburgh 2020 qui emploie un principe d'attention dénommé **CAt** qui signifie Contrastive Attention. Pour rappel, l'idée est de pouvoir pondérer l'importance des termes qui composent un document selon si ces derniers sont porteurs d'aspects. Cette méthode suppose que les noms communs les plus fréquents sont les plus susceptibles d'être le reflet d'un aspect. Cette hypothèse s'apparente avec notre observation où les termes les plus fréquents au sein des différents groupes de documents catégorisés par un aspect particulier étaient généralement des noms communs. La Contrastive Attention utilise les noms communs les plus fréquents dits « candidats ». Chaque terme du document aura un score de similarité par rapport à chacun de ces candidats. Son poids d'attention est la somme à travers ses scores redistribués par rapport à tous les termes du document via la fonction softmax. La mesure de similarité choisie est le RBF kernel (14), signifiant Radial Basis Function kernel dont ses valeurs sont comprises entre 0 et 1. Une valeur de 1 indique une forte similarité sémantique. Ce principe d'attention permet de se passer du besoin d'éliminer les stopwords. En effet, ces derniers sont ignorés car leurs poids d'attention sont proches de 0. Les poids d'attention permettent de réaliser une moyenne pondérée de l'ensemble des termes du document

afin d'obtenir une représentation de ce dernier. Il est possible d'attribuer un aspect à un document en comparant les scores de similarités entre la représentation du document et d'un aspect (tous partagent le même espace d'encodage).

Nous allons suggérer des améliorations potentielles pour l'algorithme CAt, à savoir :

- détecter la présence ou l'absence d'aspects,
- améliorer la recherche de candidats,
- améliorer la similarité entre documents et labels,
- proposer une meilleure gestion des termes peu pertinents.

### 4.8.4 Détection de la présence ou de l'absence d'aspect

La méthode basée sur CAt suppose que chaque document correspond nécessairement à un des aspects sélectionnés. Cette hypothèse n'est pas forcément vraie dans notre cas. Un document peut aborder l'aspect du prix sans que l'on souhaite y accorder de l'importance par exemple. Pour pallier cette limitation il est possible de réaliser une classification binaire séparée pour chaque aspect. Ainsi, pour un document donné, un score de similarité pourra être estimé pour chacun des aspects. S'il dépasse un seuil, l'aspect est abordé.

### 4.8.5 Recherche de candidats

Les candidats sont des facteurs importants qui déterminent les poids d'attention. Il est donc crucial de les choisir convenablement. CAt se repose sur la fréquence des noms communs au sein du corpus étudié. Les auteurs précisent qu'il est possible de choisir d'autres termes.

La figure 44 montre que la répartition des aspects parmi les noms communs les plus fréquents. Les aspects ont été labélisés à la main pour avoir : "food", "service", "ambience" et "other". Nous observons que les classes au sujet du service et de l'ambiance sont peu représentées. Cela peut affaiblir la reconnaissance de ces derniers aspects. De plus, il est possible de voir que la classe "other" occupe une proportion non négligeable et devient la deuxième classe la plus importante. Les termes rangés dans cette dernière classe apportent peu d'information ce qui peut détériorer la qualité de la classification.

On peut envisager des termes plus centrés sur les aspects et une répartition plus équilibrée. Il est possible de générer automatiquement les candidats au lieu de les récupérer depuis le corpus. La génération se fait à partir du modèle Word2Vec et des aspects. Pour

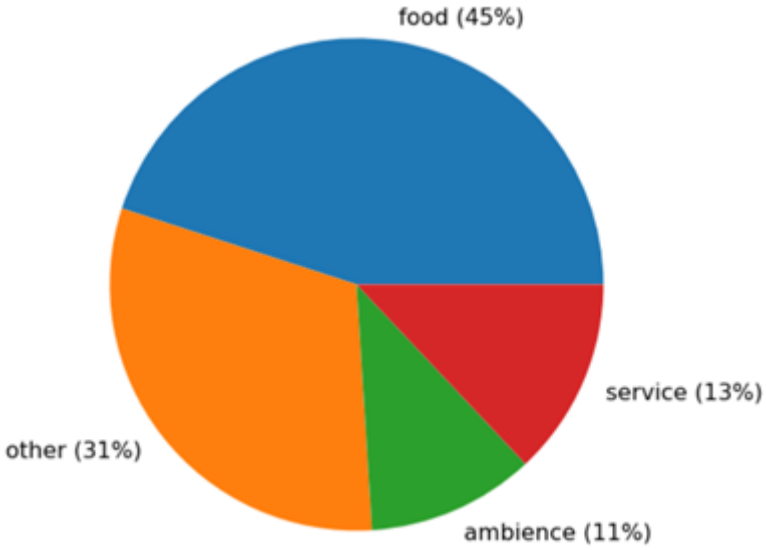


FIGURE 44 – Proportion des aspects parmi les candidats
Les candidats correspondent aux termes retrouvées dans les 200 noms communs les plus fréquents du jeu d'entraînement. Il s'agit des candidats trouvés dans l'algorithme CAt.

chaque aspect, un ensemble de termes est généré en choisissant les mots les plus semblables selon le Word2Vec. La similarité se calcule par le cosinus définie par l'équation 15.

```
['food', 'cuisine', 'meals', 'sushi', 'fare', 'meal', 'pizza', 'grub', 'foods', 'bibimbop',
'service', 'waitstaff', 'staff', 'communication', 'sevice', 'services', 'attitudes',
'serivce', 'workmanship', 'experience', 'ambience', 'ambiance', 'atmosphere', 'decor',
'environment', 'vibe', 'interior', 'setting', 'aesthetic', 'presentation']
```

FIGURE 45 – Exemple de 30 candidats générés
Génération en choisissant les termes les plus similaires aux aspects "food", "service" et "ambience". Ces termes ont été rassemblés en une seule liste.

### 4.8.6 Similarité avec les labels

Nous cherchons à fiabiliser le rapprochement aspect-document. La figure 46 illustre une projection en deux dimensions des embeddings de Word2Vec. **L'encodage des trois aspects est représenté en noir**. Le **point rouge** est un exemple tiré du jeu de test : « the bread was top notch ». Ce document fait clairement référence à l'aspect « food ». Dans l'espace d'origine, la similarité entre la phrase et le mot food est de 0.148 tandis que la similarité avec l'aspect "ambience" vaut 0.131. La différence est très faible. Cela est dû au fait que l'expression « top notch » est généralement utilisée pour qualifier l'ambiance.

Comme moyen de renforcement, il est possible d'associer une liste de termes pour chaque aspect. Cela forme un thésaurus. La figure 47 illustre un exemple de thésaurus

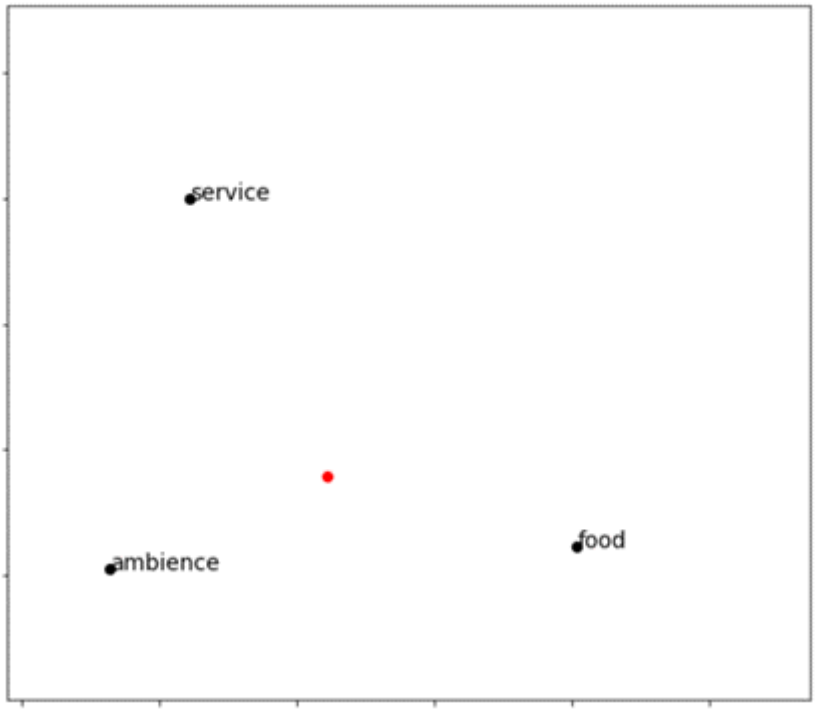


FIGURE 46 – Représentation des aspects et de la phrases test en deux dimensions. Ce plan est l'espace de projection qui résulte d'une analyse par composantes principale sur un sous-ensemble de mots (en utilisant leur représentation vectorielle donnée par le modèle Word2Vec).

généré avec les trois aspects "food", "service" et "ambience". Chaque aspect a été peuplé par les termes les plus similaires selon le modèle Word2Vec.

```
{
  'ambience': 'ambience, ambiance, atmosphere, decor, environment, vibe, '
              'interior, decoration, setting, scenery, atmoshere, surroundings, '
              'vibes, aesthetic, decorations, furnishings, layout, chic, '

  'food': 'food, cuisine, meals, fare, sushi, meal, ming, pizza, grub, foods, '
          'tapas, cusine, cuisines, fusion, desta, authenticity, influences ',

  'service': 'service, waitstaff, staff, communication, attitudes, sevice, '
             'workmanship, services, serivce, timing, antjuan, servers, '
             'waiters, waitresses, bartenders, staffs, hostesses, baristas '
}
```

FIGURE 47 – Exemple de thésaurus pour les aspects.
Ces termes ont été trouvé en utilisant les mots les plus similaires aux aspects dont les représentations sont données par le modèle Word2Vec.

Sur la figure ci-dessous, le même exemple a été repris. Chaque point de couleur correspond à un mot du thésaurus. La couleur permet de distinguer la classe d'appartenance. Nous pouvons observer que le point rouge est entouré par des mots de la classe food et

que la classification pourra être plus fiable si elle se base sur les points qui l'environnent. **Ainsi, le document est classé selon le terme t le plus proche. L'aspect associé à t sera la classe du document.**

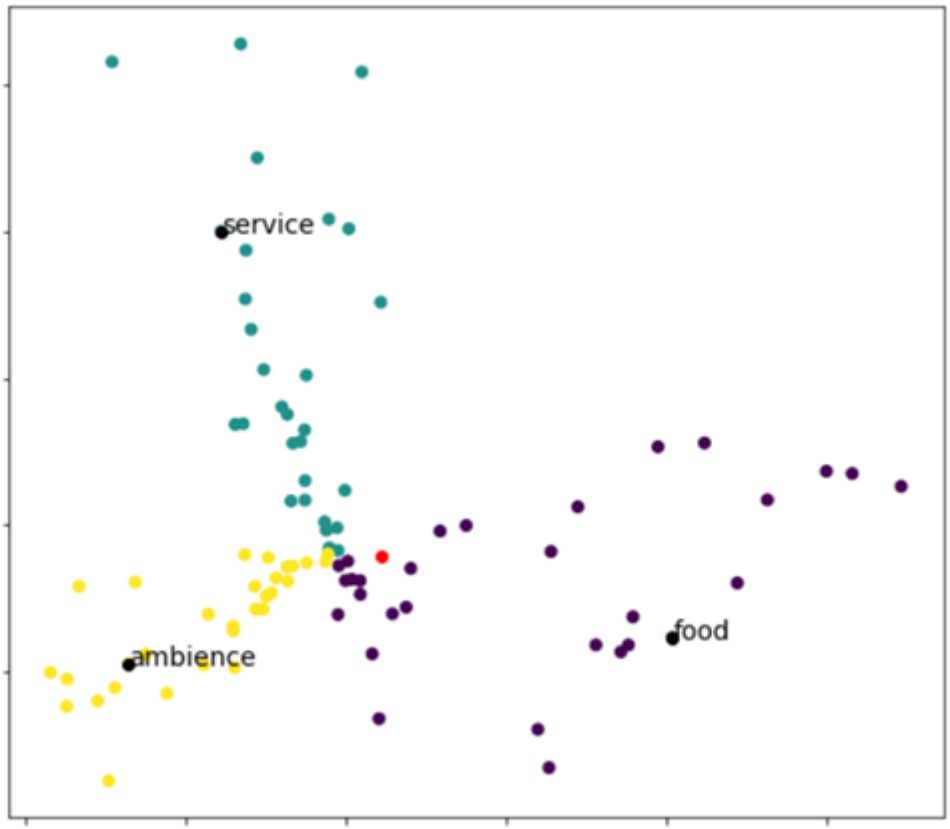


FIGURE 48 – Représentation des aspects et de la phrases test en deux dimensions
Ce plan est l'espace de projection qui résulte d'une analyse par composantes principale sur un sous-ensemble de mots (en utilisant leur représentation vectorielle donnée par le modèle Word2Vec). Cette figure inclut les mots du thésaurus. Les couleurs permettent de séparer ces derniers selon l'aspect auquel ils appartiennent.

### 4.8.7 Gestion des termes peu pertinents

Dans la méthode CAt, les documents n'ont pas à vocation d'être débarrassés des termes non-pertinents tels que les stopwords. En effet, le mécanisme d'attention permet en théorie d'ignorer les termes peu importants. Toutefois, ces derniers ajoutent du bruit qui tend à détériorer les performances car les poids d'attention ne sont pas toujours nuls. Par exemple, la phrase « i have to mention that the steak was excellent » contient six termes non pertinents pour trois porteurs de sens (voir un seul porteur d'aspect). Nettoyer cette phrase mènerait à obtenir la suivante : « mention steak excellent ». **Ainsi, il est préférable de retirer les termes non importants tels que les pronoms, déterminants, etc.**

Il est possible d'envisager de supprimer les adjectifs et verbes et de ne garder que les

| Méthodes | F1-score |
|---|---|
| **CAt** | 0.742 |
| **CAt sans stopwords** | 0.747 |
| **CAt + noms communs seulement** | 0.732 |
| **CAt + changement des candidats** | 0.730 |
| **CAt + similarité avec les labels** | **0.780** |
| **CAt + combinaison** | 0.776 |

TABLE 7 – Récapitulatif des scores engendrés par les améliorations suggérée pour CAt. Nous avons choisi le contexte d'une classification entre les trois labels "food", "service" et "ambience" pour plus de ressemblance avec le contexte d'évaluation posé par les auteurs de CAt.

noms communs (porteurs d'aspect). En effet, ces derniers suffiraient pour retrouver l'aspect.

#### 4.8.8 Expérimentations avec les améliorations potentielles citées

Afin d'évaluer ces potentielles améliorations, nous allons nous placer dans la même problématique que CAt, à savoir classifier les phrases selon trois étiquettes : "food", "service" ou "ambience". Étant donné qu'il s'agit d'une classification a plus de deux classes, **la macro F1-score** sera employée.

$$Macro\ F1 = \frac{1}{N}\sum_{i=0}^{N} F1_i \tag{20}$$

Le tableau suivant résume les performances des différents apports à CAt. On note :

- **Sans stopwords** : retirer les stopwords. Les adjectifs et verbes ne sont pas supprimés.
- **Nom commun seulement** : garder uniquement les noms communs.
- **Changement de candidats** : définir les candidats à partir du Word2Vec en choisissant les termes les plus similaires aux aspects.
- **Similarité avec les labels** : la représentation vectorielle d'un document est comparée aux termes du thésaurus. Ce dernier est généré à partir du Word2Vec. Dans cette expérience, les comparaisons se basent sur neuf termes au total.

Le tableau 7 montre les performances de ces améliorations. Nous constatons que garder uniquement les noms communs fait diminuer le score de 0.10. De plus, changer la matrice d'attention détériore légèrement le résultat final. En effet, il est possible que les verbes et adjectifs soient utiles pour reconnaître l'aspect la plupart du temps. Par exemple, le mot

« delicious » est plus proche de l'aspect concernant la nourriture que du service. Ainsi, conserver ces termes peut être bénéfique. **Néanmoins, enrichir les labels afin d'avoir plus de comparaisons possibles a permis d'augmenter les performances.**

Au niveau du nombre de termes à générer pour réaliser la comparaison entre documents et aspects, noté $k$, il est possible d'améliorer les performances en faisant varier $k$. La figure 49 montre que le maximum est atteint pour k=21 et k=24 pour un F1-score de 0.7992.

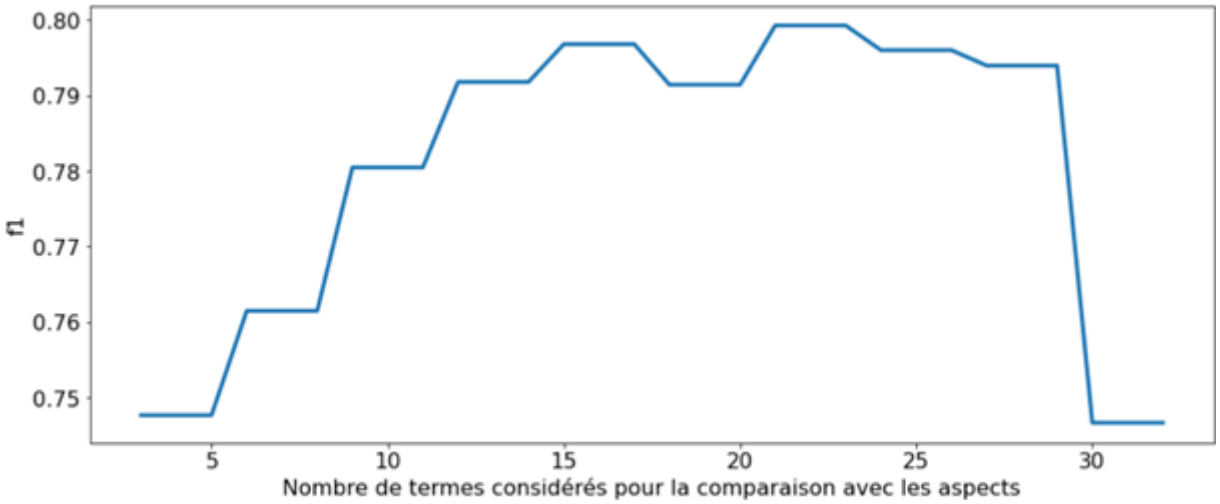


FIGURE 49 – Évolution du f1-score face au nombre de termes $k$ considérés dans la classification en aspects

Ainsi, nous pouvons valider l'intérêt de retirer les stopwords tels que les articles et pronoms. De plus, enrichir les labels avec un thésaurus permet d'obtenir des résultats plus exacts.

### 4.8.9 Algorithme retenu

Afin de détecter la présence d'un aspect au sein d'un fragment de texte, l'algorithme CAt servira de base dans lequel est incorporé les améliorations citées :

- Suppression des stopwords,
- Enrichissement des labels avec un thésaurus.

Comme modification nécessaire, la problématique change entre le contexte posé par les auteurs de CAt et le contexte sur lequel nous travaillons. En effet, notre problématique est de pouvoir repérer la mention d'un aspect. Ainsi, **l'algorithme consistera à réaliser une détection indépendante et centrée sur un aspect à la fois.** À l'échelle d'un aspect, cela revient à faire une classification binaire pour savoir si le document fait référence ou non à l'aspect ciblé.

Le calcul de la similarité est basé sur la similarité du cosinus. Cette dernière est normalisée pour borner ses valeurs entre 0 et 1 dans le but de modéliser une probabilité pour le résultat final. Le calcul de la similarité entre deux vecteurs $u$ et $v$ est la suivante :

$$sim(u, v) = max(cos(u, v), 0) \tag{21}$$

Pour un aspect choisi, le processus de détection au sein d'un document contient une phase d'initialisation et une phase de détection. Dans la phase d'initialisation, il est question de constituer la matrice d'attention $A$ en retrouvant les noms communs les plus répandus dans le corpus d'entrée. Il s'agit des candidats, dont l'ensemble est noté $C$. De plus, une liste de termes similaires à l'aspect est établie en se basant sur le modèle Word2Vec ce qui permet d'avoir une matrice d'embeddings de l'aspect $L$ (les embeddings se lisent en colonne).

La phase de détection est explicitée ci-dessous :

1. Le document est retiré de ces stopwords.
2. Chaque terme du document est retranscrit dans sa représentation vectorielle ce qui résulte d'une séquence d'embeddings $S$.
3. À partir de la séquence d'embeddings $S$ et de la matrice d'attention $A$ sont calculés les poids d'attention pour chaque terme du document. Pour un mot $w$, on applique l'équation 13. On obtient des poids provenant de chacun des candidats. Par exemple, le mot $w$, pour un candidat $c$ a reçu le poids $att_w^c$. Afin d'avoir un unique poids pour un seul mot, on effectue la somme des poids donnant $a_w$. Les poids seront réajustés par une fonction softmax ce qui donne les poids finaux $\alpha$. Ainsi,

$$a_w = \sum_{c \in C} att_w^c \tag{22}$$

$$\alpha_w = softmax_a(a_w) \tag{23}$$

$$softmax_a(a_w) = \frac{exp(a_w)}{\sum_j exp(a_j))} \tag{24}$$

4. La représentation vectorielle du document est obtenue en appliquant la moyenne des embeddings de la séquence pondérée par les poids d'attention. On obtient une représentation du document $d = \sum_{w \in S} att_w.w$

5. On calcule la similarité entre le document et les colonnes de la matrice de l'aspect $L$ avec la formule 21.
6. La probabilité que le document aborde l'aspect est modélisée par le maximum des similarités.

La dernière étape consiste à comparer la probabilité avec un seuil de référence fixé. Ce seuil permet de décider si la probabilité que le commentaire aborde l'aspect est suffisamment grande pour considérer que l'aspect a été mentionné. La figure 50 résume le processus de détection d'aspects.

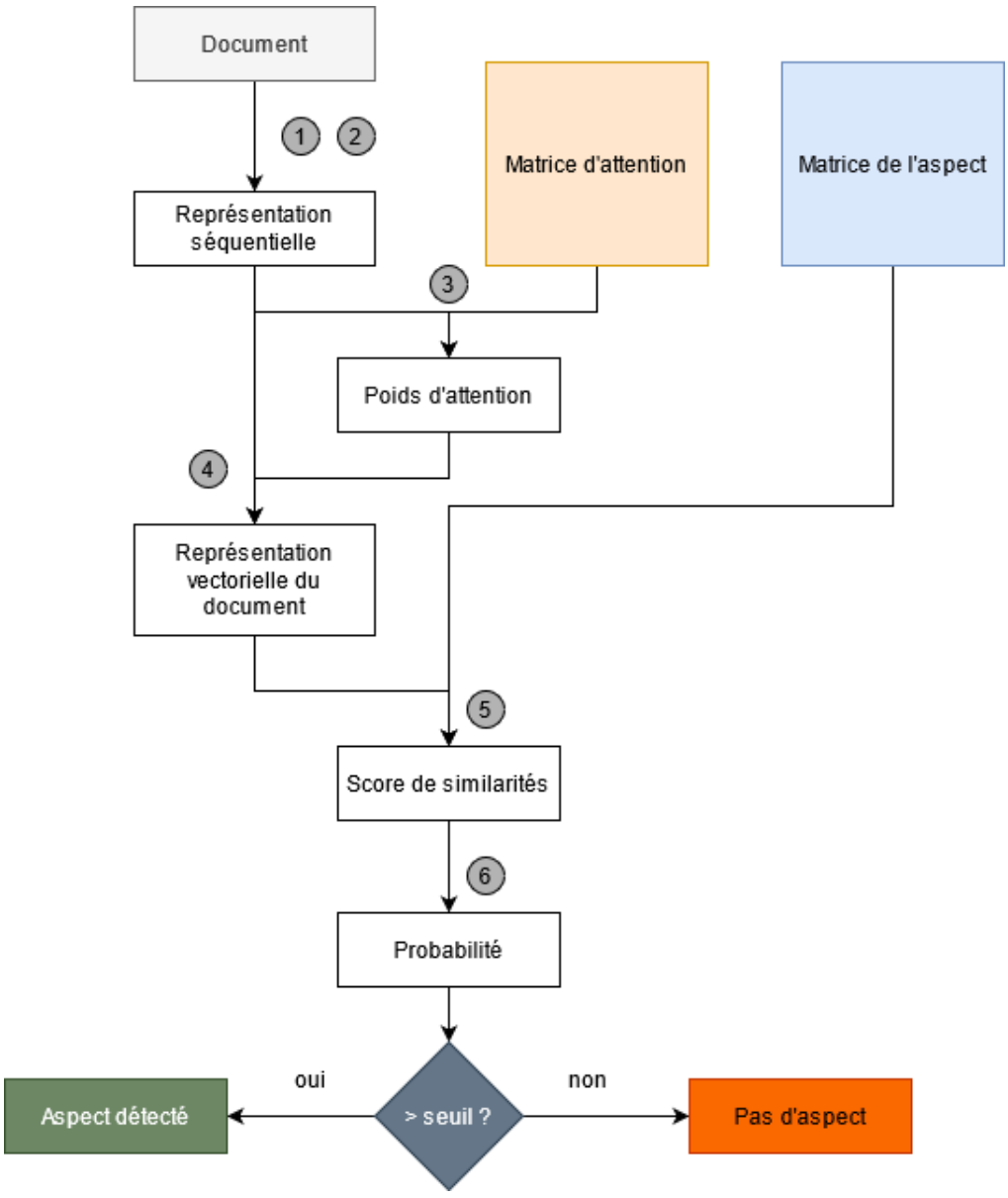


FIGURE 50 – Processus de détection d'aspect.

### 4.8.10 Résultats et analyse des erreurs

Les expériences ont été réalisées avec une valeur de gamma à 0.03 comme suggéré par les auteurs de CAt et 100 candidats. La figure 51 montre l'évolution du F1-score par aspect selon le seuil à partir duquel on considère que l'aspect a été détecté. Un seuil de 0 signifie que tous les documents seront considérés comme abordant l'aspect. Nous voyons que les aspects liés au service et à l'ambiance atteignent un pic pour un seuil aux alentours de 0.47. Leur F1-score maximal est respectivement de 0.65 et 0.57. Cela représente un gain de 0.28 et de 0.30 par rapport à un algorithme prédisant toujours « vrai » pour la détection d'aspects.

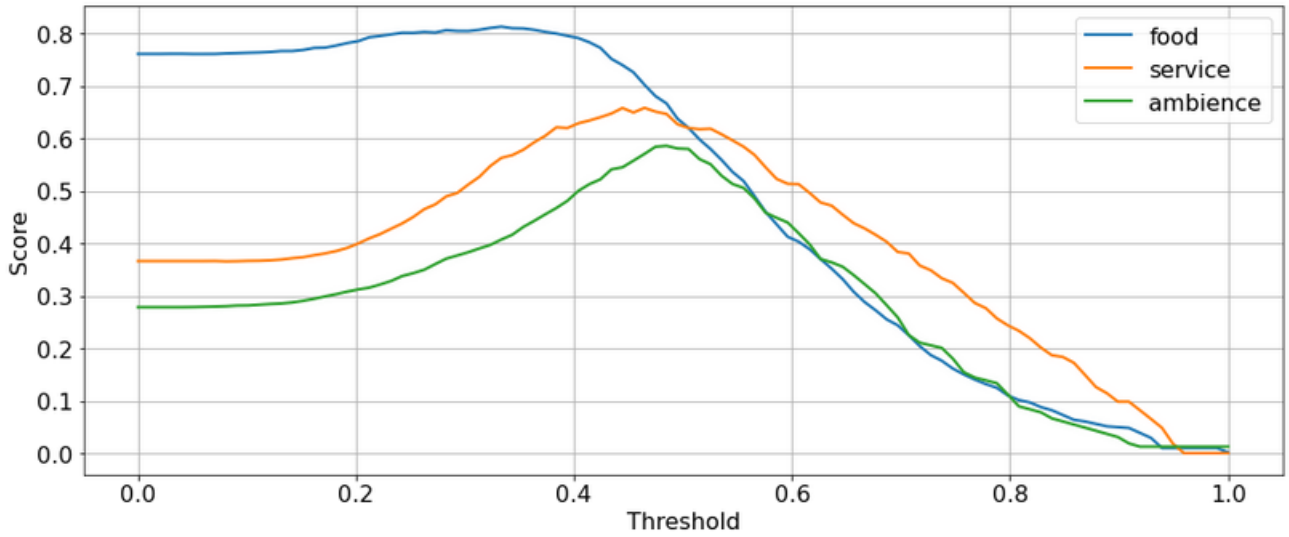


FIGURE 51 – Évolution du F1-score dans la détection d'aspects

Le gain est moins prononcé au sujet de l'aspect sur la nourriture. En effet, le pic est atteint à un seuil de 0.33 pour un F1-score de 0.80 ce qui représente un gain de 0.05 par rapport à un algorithme prédisant constamment « vrai ». La figure 52 montre un échantillon de faux positifs commis par l'algorithme.

Il est possible de distinguer deux types d'erreur :

- La forte corrélation entre les mots « restaurant » et « food ». En effet, ce premier terme est récurrent dans l'échantillon. Par conséquent, il se retrouve parmi les candidats.
- L'emploi trompeur d'un terme relié à l'aspect « food ». Par exemple, la ligne 8 montre un document abordant le service en se plaignant de la lenteur de ce dernier. Le mot « treat » est employé mais ne constitue pas le sujet de la phrase.

Pour des seuils fixés à 0.5, les performances sont résumées dans le tableau 8. Pour obtenir ces résultats, le terme "restaurant" a été retiré du corpus. Nous pouvons observer un gain

| | A | B | |
|---|---|---|---|
| 1 | **clean_sentence** | **aspects** | **proba_food** |
| 2 | service experience friendly good | service | 0.711888783454161 |
| 3 | decor design contemporary japanese style restaurant | ambience | 0.6116779717371774 |
| 4 | beautiful atmosphere perfect drink appetizer | ambience | 0.5297844926285082 |
| 5 | one waiter whole restaurant upstairs | service | 0.6008575518250467 |
| 6 | soon one person ask pick plate immediately | service | 0.5007157860278901 |
| 7 | crowd old restaurant cramp old school charm | ambience | 0.5146606945590246 |
| 8 | wait three hour entree treat well | service | 0.6425535599650026 |
| 9 | peak time restaurant overcrowded table uncomfortably close | ambience | 0.5966915277103647 |
| 10 | restaurant quiet intimate | ambience | 0.822085862957139 |
| 11 | atmosphere take place many dream | ambience | 0.6199426473819961 |
| 12 | visit new york city friend discover really warm invite restaurant | ambience | 0.524700083896172 |
| 13 | great place relax enjoy dinner | ambience | 0.731551594675037 |
| 14 | beautifully design dreamy egyptian restaurant scene night | ambience | 0.6166516180443649 |

FIGURE 52 – Échantillon de faux positifs dans la détection d'aspects

| Aspects | Precision | Recall | F1 | Gain (F1) |
|---|---|---|---|---|
| **Food** | 0.91 | 0.46 | 0.62 | -0.14 |
| **Service** | 0.80 | 0.50 | 0.62 | +0.25 |
| **Ambience** | 0.79 | 0.45 | 0.57 | +0.30 |

TABLE 8 – Performance de la détection d'aspects à un seuil $p = 0.5$
Le gain représente la différence entre le f1-score à un seuil fixé à 0.5 et un détecteur prédisant toujours "vrai" pour tout aspect.

de performance pour les aspects "service" et "ambience". Néanmoins, la détection pour l'aspect "food" est peu performant. La précision montre que la détection est pertinente. En revanche, le recall plus faible indique qu'une portion considérable de documents abordant un aspect n'a pas été détectée. Nous validons le modèle pour sa haute précision.

# 5 Applicabilité de la solution

## 5.1 Démonstration d'un produit

À ce stade, il est possible de construire la synthèse d'opinions en enchaînant la segmentation des sentiments avec la détection d'aspects. Les fragments de texte issus de la segmentation des phrases isolant le sentiment sont analysés indépendamment lors de la détection d'aspects. Le croisement sentiment-aspect permet de déterminer l'opinion vis-à-vis de l'aspect mentionné. **Les paires sentiment-aspect ne sont comptabilisées qu'une seule fois dans un seul commentaire.**

Ainsi, nous pouvons construire une solution complète de synthèse d'opinions. Afin d'illustrer ce propos, nous pouvons scraper des commentaires textuels d'un restaurant afin de constituer le jeu de données à analyser. La synthèse d'opinions sera appliquée à

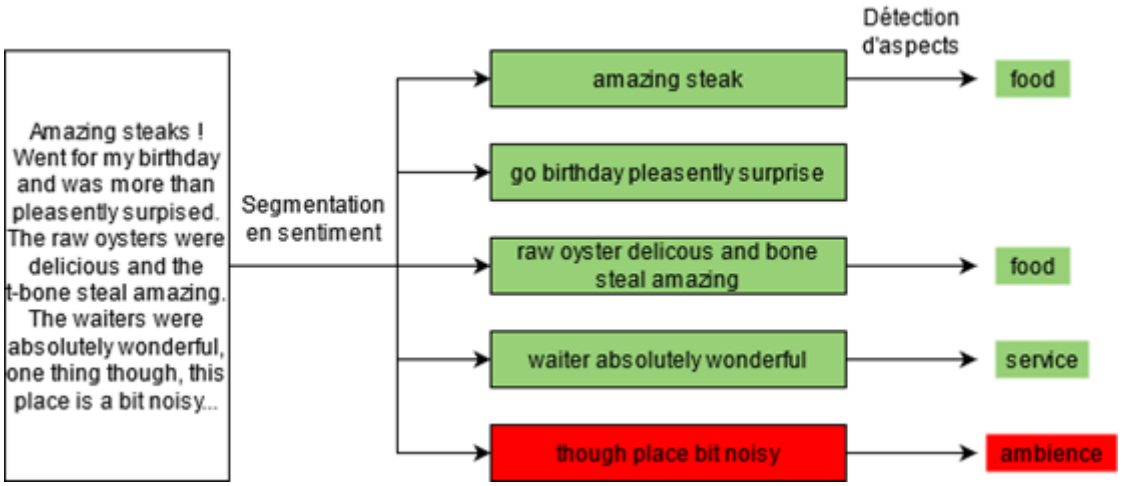


FIGURE 53 – Exemple montrant le déroulé du processus de synthèse

ces données. Des visualisations pourront être construites à partir des résultats. La figure 54 illustre le système mentionné.

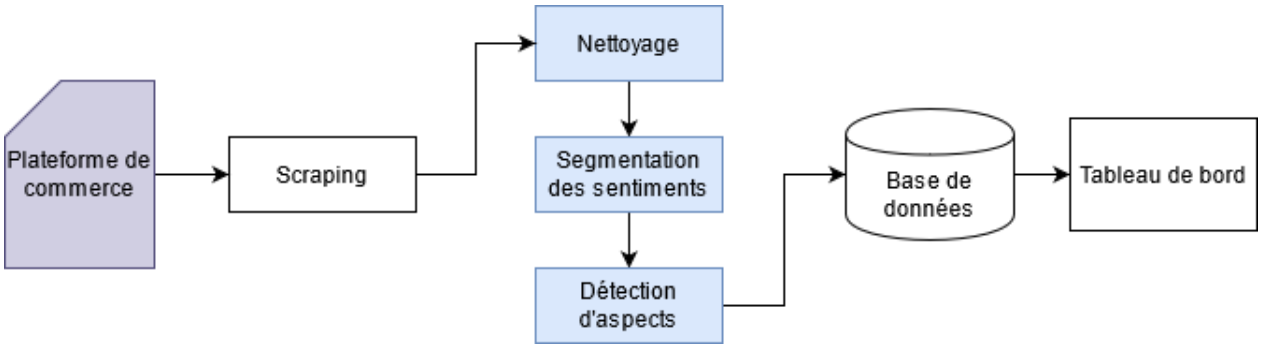


FIGURE 54 – Schéma d'un produit utilisant notre algorithme

Les résultats alimentent un tableau de bord permettant de visualiser l'opinion exprimée dans les commentaires. La figure 55 montre un exemple de tableau de bord récupérant les données traitées par notre algorithme et affichant des diagrammes circulaires et en bâtons pour synthétiser l'opinion par aspect visuellement.

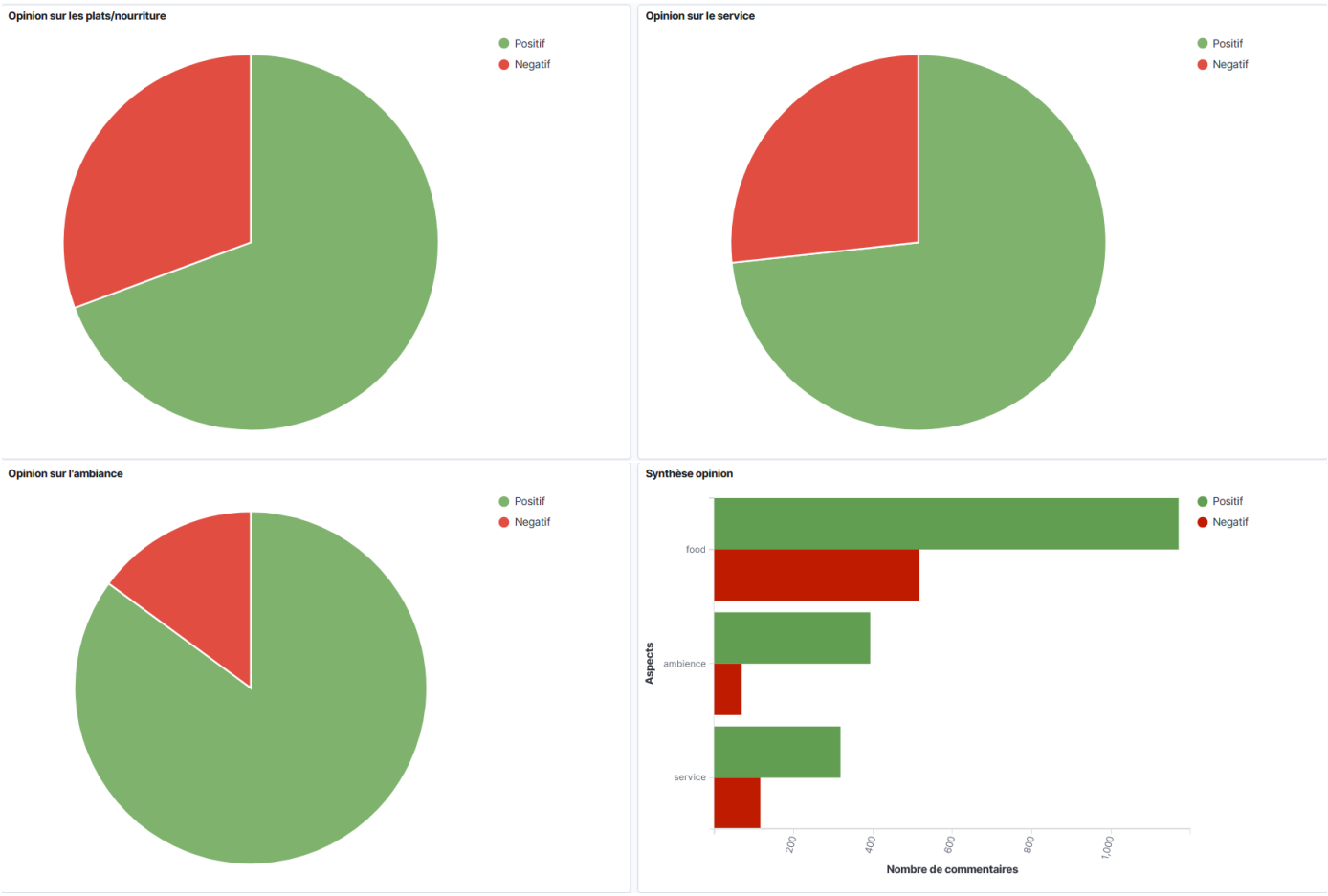


FIGURE 55 – Tableau de bord réalisé synthétisant des données de Yelp
Réalisé avec Kibana.

## 5.2 Généralisation

Il est possible de **généraliser la solution à d'autres domaines**. Il suffit d'entraîner un modèle d'embedding tel que Word2Vec dans le domaine étudié (cinéma, produits cosmétiques...). La listes des aspects à considérer est un paramètre à définir. À ce stade, il est possible de relancer le processus d'entraînement pour la segmentation des sentiments et l'algorithme pour la détection d'aspects.

En outre, la segmentation des sentiments permet d'estimer un indice d'opinions positifs et négatifs **plus précis** qu'une simple classification du commentaire entier.

Ce système de synthèse est utile pour connaître rapidement l'opinion exprimée et filtrer le contenu textuel pour ne retenir que les aspects pertinents. Cela peut servir dans l'analyse des verbatims pour simplifier et raccourcir le temps de travail permettant de comprendre l'opinion. La combinaison avec **un tableau de bord permet d'automatiser l'affichage des visualisations.** Ces dernières peuvent également être au service des utilisateurs qui souhaitent avoir rapidement un aperçu des avis exprimés sur un produit dans

les sites d'e-commerce par exemple. En effet, **les éléments graphiques permettent de rassembler toute l'information pertinente pour être lue et comprise efficacement**. Cela évite le travail chronophage de devoir examiner un échantillon de commentaires et chercher les aspects intéressants manuellement. Ainsi, un tel système peut être incorporé dans un outil d'aide à la décision.

# 6 Discussion sur les intérêts, limites et pistes d'amélioration

Le principal défi posé pour l'analyse des sentiments par l'aspect réside dans le moyen de relier le sentiment à l'aspect et détecter l'aspect de manière non-supervisée. Pour relever ce défi, ce travail montre une approche qui se base sur le croisement sentiment-aspect. L'idée de segmenter le commentaire pour isoler le sentiment puis d'y repérer les aspects n'a pas été, à ma connaissance, traité dans la littérature ce qui apporte une nouvelle manière d'approcher le problème. De plus, cette méthode utilise le même mécanisme d'attention que CAt (Tulkens et Cranenburgh 2020) ce qui rend les résultats interprétables en examinant les termes qui réagissent le plus avec les aspects (celles qui ont mérité un grand poids d'attention).

Toutefois, les commentaires peuvent être variés et contenir des passages sans sentiment ni aspect. Il serait donc intéressant d'ajouter **le label « neutre »** dans l'analyse des sentiments pour affiner les résultats. Quant à la détection d'aspects, elle a été réalisée de manière à ignorer les passages sans aspect.

De plus, il est très probable qu'on puisse réduire la quantité d'erreurs commises dans le système de synthèse en améliorant la génération de données et l'architecture du réseau de neurones (pour l'analyse des sentiments) et en cherchant de meilleurs modèles de représentation vectorielles ou des paramètres plus adaptés par exemple.

Par ailleurs, la méthode suppose que les aspects soient clairement évoqués dans chacune des phrases ou bien soient complètement absents. Par exemple, une reprise d'un aspect par un pronom fera perdre de l'information cruciale. **Le modèle d'embedding utilisé n'est pas contextualisé**. Il peut être trompé par des termes qui ne sont pas sujet de la phrase comme dans l'exemple : « I waited one hour to get my food » où l'aspect est le service tandis que le système aurait tendance à suggérer la nourriture.

De plus, la détection d'aspect repose fortement sur la capacité à repérer correctement les noms communs. Une erreur dans de POS tagging au niveau des noms va détériorer le mécanisme d'attention (recherche de candidats) et ainsi la représentation du document. Des erreurs peuvent se glisser en raison des limitations du modèle de POS-tagging et de la qualité de la rédaction du commentaire (fautes de frappe, syntaxes, orthographe...). **La mise en qualité du texte est donc une étape primordiale qui représente un coût pour la complexité de sa mise en place et pour son temps de calcul** si ce système en venait à être industrialisé.

Les modèles de langue plus récents tels que **RoBERTA et XLNet** ont été entraînés pour interpréter les détails du langage tels que les ponctuations et les fautes, ce qui nécessite peu de nettoyage en amont. C'est pourquoi ces modèles de langues sont des solutions intéressantes pour remplacer le modèle d'embedding actuel. En plus de proposer des représentations contextualisées, ils donnent généralement les meilleures performances à ce jour dans le domaine de l'analyse des sentiments [4] et peuvent servir de remplacement au réseau BiLSTM actuel.

Néanmoins, les coûts d'entraînement et de déploiement de ces modèles peuvent être **élevés** étant donné les dimensions mémoires et la latence de ces derniers. Un axe de recherche se concentre sur **la compression des modèles de langue** dont le défi est de réduire leurs besoins en mémoire et en temps de calcul sans perdre les performances initiales. Nous pouvons citer les travaux sur MobileBERT qui présentent une version compacte du modèle BERT d'origine.

Une meilleure représentation du texte est donc un enjeu majeur dans l'amélioration de ce système. Les modèles de langues précédentes, s'ils ne sont pas suffisants en tant que tel, ne peuvent être adaptés au domaine que s'ils sont entraînés sur une tâche prétexte. Cela peut augmenter les performances dans la détection d'aspects mais implique des données labélisées. De manière générale, des techniques auto-supervisées (self-supervised) ont été développées parmi lesquelles nous pouvons citer **les méthodes d'apprentissage contrastées** (contrastive self-supervised learning). Ces dernières portent leurs racines dans la Computer Vision. L'idée est d'entraîner un réseau de neurones pour rapprocher deux représentations jugées préalablement similaires et repousser celles jugées différentes. Les représentations sont issues de l'augmentation d'une même image dans le cas d'un rapprochement. La **data augmentation dans le TALN** peut se faire en remplaçant des termes par

4. Source : NLP Progress

leurs synonymes ou encore par back translation : traduire le texte dans une langue puis à nouveau dans la langue d'origine afin que le bruit accumulé lors des deux traductions produise deux textes différents mais avec une même sémantique (voir la figure 56). Ces méthodes de représentation vectorielle pourraient mieux capturer les nuances sémantiques qu'un modèle Word2Vec.

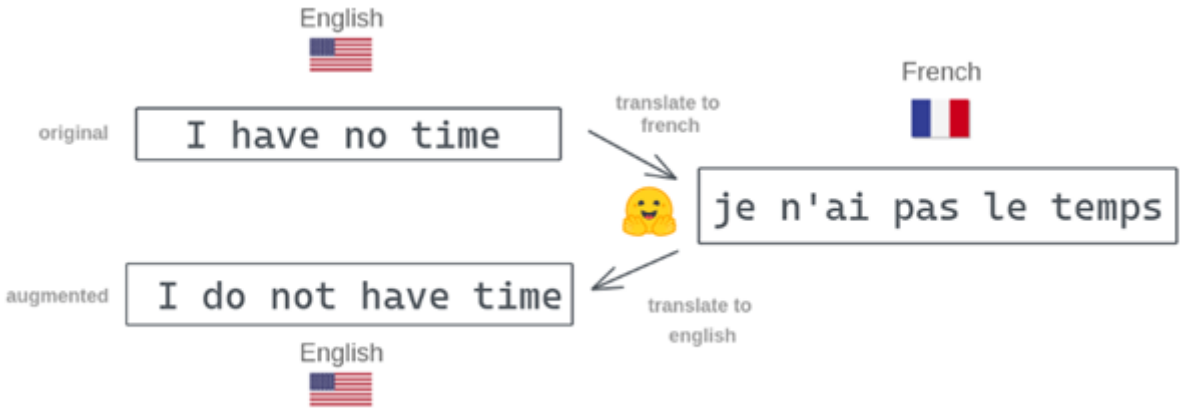


FIGURE 56 – Illustration du concept de la back translation sur une phrase
La traduction s'est effectuée de l'anglais vers le français, puis du français vers de l'anglais. Nous pouvons observer une différence entre la phrase initiale et la phrase finale après deux traductions. Source : https ://amitness.com/back-translation/

En outre, cette méthode demande une connaissance du domaine car les aspects sont à renseigner par son utilisateur. En effet, **l'identification automatique des aspects** ne fait pas partie de la problématique. Pour faciliter l'usage du système, il serait intéressant d'ajouter cette étape. Avec une représentation pertinente des documents, étudier la possibilité de regrouper ces derniers en clusters pourrait aider à identifier les aspects. HE et al. 2017 ont proposé une solution pour regrouper les documents en sous-aspects. Néanmoins, la liaison entre ces derniers et les aspects recherchés a dû être réalisée manuellement.

# Conclusion

Ce travail a permis de développer une approche pour synthétiser l'opinion exprimée dans un corpus. Nous avons vu comment l'analyse des sentiments par aspect permet de résoudre ce problème en ciblant des aspects particuliers du sujet afin d'en exhiber les sentiments exposés.

Les données émises du SemEval et de Citysearch ont constitué un point de départ. Les analyses des données ont permis de formuler des hypothèses sur la construction des commentaires. En effet, nous avons observé que la formulation des sentiments dans une phrase permettait de séparer cette dernière en segments de texte afin d'isoler l'opinion. Cela constitue la première étape de la synthèse.

En modélisant le problème en une classification de tokens, un réseau BiLSTM a été entraîné dans le but de pouvoir segmenter les phrases et estimer le sentiment. Les segments de texte sont réutilisés pour extraire les aspects abordés de manière non-supervisée. Nous avons déterminé les limites de l'algorithme CAt pour proposer des améliorations qui ont pu augmenter ses performances dans les données utilisées. Le croisement sentiment-aspect permet de déterminer l'opinion vis-à-vis de l'aspect en question. Afin d'obtenir une synthèse, la présentation des résultats sous la forme de graphiques permet de distinguer les aspects ainsi que les proportions d'avis positifs ou négatifs au sein de ces aspects.

Pour finir, nous avons constaté les limites de cette méthode et abordé les éventuelles améliorations qui pourraient rectifier les défauts de ce système. En effet, la représentation vectorielle des textes est un facteur central dans le calcul des similarités, dans lequel la méthode présentée y dépend fortement.

Après l'émergence du Web, la multiplication des échanges et publications sur les réseaux sociaux se sont renforcées à l'échelle mondiale. La langue est devenue un sujet capital dans le traitement automatique du langage naturel. En effet, des plateformes d'e-commerce ont déjà rassemblé des avis à travers le monde. Des recherches visant à développer des représentations multilingues ont permis d'ouvrir une nouvelle voie pour briser les barrières imposées par les langues. Pour mieux adapter la méthode au contexte actuel, il serait intéressant d'étendre la solution en considérant l'intégralité des commentaires quel que soit la langue employée. Cela permettra d'agrandir son périmètre d'application à un cercle international.

# Références


[AA20] V.S. Anoop et S. Asharaf. "Aspect-Oriented Sentiment Analysis : A Topic Modeling-Powered Approach :" in : *Journal of Intelligent Systems* 29.1 (2020), p. 1166-1178. doi : doi:10.1515/jisys-2018-0299. url : https://doi.org/10.1515/jisys-2018-0299.

[BCB16] Dzmitry Bahdanau, Kyunghyun Cho et Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. 2016. arXiv : 1409.0473 [cs.CL].

[BE10] Samuel Brody et Noémie Elhadad. "An Unsupervised Aspect-Sentiment Model for Online Reviews". In : *NAACL*. 2010.

[BNJ03] David M. Blei, Andrew Y. Ng et Michael I. Jordan. "Latent dirichlet allocation". In : *J. Mach. Learn. Res.* 3 (2003), p. 993-1022. issn : 1532-4435. doi : http://dx.doi.org/10.1162/jmlr.2003.3.4-5.993. url : http://portal.acm.org/citation.cfm?id=944937.

[Boj+17] Piotr Bojanowski et al. *Enriching Word Vectors with Subword Information*. 2017. arXiv : 1607.04606 [cs.CL].

[Dev+19] Jacob Devlin et al. *BERT : Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv : 1810.04805 [cs.CL].

[GCR17] Aitor García-Pablos, Montse Cuadros et German Rigau. *W2VLDA : Almost Unsupervised System for Aspect Based Sentiment Analysis*. 2017. arXiv : 1705.07687 [cs.CL].

[Gha+19] Erfan Ghadery et al. *An Unsupervised Approach for Aspect Category Detection Using Soft Cosine Similarity Measure*. 2019. arXiv : 1812.03361 [cs.CL].

[He+17] Ruidan He et al. "An Unsupervised Neural Attention Model for Aspect Extraction". In : *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*. Vancouver, Canada : Association for Computational Linguistics, juil. 2017, p. 388-397. doi : 10.18653/v1/P17-1036. url : https://aclanthology.org/P17-1036.

[HLQ19] Tai Hoang, Huy Le et Tho Quan. "Towards Autoencoding Variational Inference for Aspect-Based Opinion Summary". In : *Applied Artificial Intelligence* 33.9 (juin 2019), p. 796-816. issn : 1087-6545. doi : 10.1080/08839514.2019.1630148. url : http://dx.doi.org/10.1080/08839514.2019.1630148.

[Lee+17] Ji-Ung Lee et al. “UKP TU-DA at GermEval 2017 : Deep Learning for Aspect Based Sentiment Detection”. In : *Proceedings of the GSCL GermEval Shared Task on Aspect-based Sentiment in Social Media Customer Feedback*. German Society for Computational Linguistics. Berlin, Germany, sept. 2017, p. 22-29.

[Liu+19] Yinhan Liu et al. *RoBERTa : A Robustly Optimized BERT Pretraining Approach*. 2019. arXiv : 1907.11692 [cs.CL].

[MHM20] Leland McInnes, John Healy et James Melville. *UMAP : Uniform Manifold Approximation and Projection for Dimension Reduction*. 2020. arXiv : 1802.03426 [stat.ML].

[Mik+13] Tomas Mikolov et al. *Efficient Estimation of Word Representations in Vector Space*. 2013. arXiv : 1301.3781 [cs.CL].

[Mov+19] Sajad Movahedi et al. *Aspect Category Detection via Topic-Attention Network*. 2019. arXiv : 1901.01183 [cs.CL].

[NLM19] Jianmo Ni, Jiacheng Li et Julian McAuley. “Justifying recommendations using distantly-labeled reviews and fined-grained aspects”. In : *EMNLP*. 2019.

[PSM14] Jeffrey Pennington, Richard Socher et Christopher D. Manning. “GloVe : Global Vectors for Word Representation”. In : *Empirical Methods in Natural Language Processing (EMNLP)*. 2014, p. 1532-1543. url : http://www.aclweb.org/anthology/D14-1162.

[Qiu+11] Guang Qiu et al. “Opinion Word Expansion and Target Extraction through Double Propagation”. In : *Computational Linguistics* 37.1 (2011), p. 9-27. doi : 10.1162/coli_a_00034. url : https://aclanthology.org/J11-1002.

[Sch+18] Martin Schmitt et al. *Joint Aspect and Polarity Classification for Aspect-based Sentiment Analysis with End-to-End Neural Networks*. 2018. arXiv : 1808.09238 [cs.CL].

[Sid+14] G. Sidorov et al. “Soft Similarity and Soft Cosine Measure : Similarity of Features in Vector Space Model”. In : *Computación y Sistemas* 18 (2014).

[SS17] Akash Srivastava et Charles Sutton. *Autoencoding Variational Inference For Topic Models*. 2017. arXiv : 1703.01488 [stat.ML].

[TBH19] Thang Tran, Hung Ba et Van-Nam Huynh. “Measuring Hotel Review Sentiment : An Aspect-Based Sentiment Analysis Approach”. In : *Integrated Uncertainty in Knowledge Modelling and Decision Making*. Sous la dir. d’Hirosato Seki

et al. Cham : Springer International Publishing, 2019, p. 393-405. ISBN : 978-3-030-14815-7.

[TC20] Stéphan Tulkens et Andreas van Cranenburgh. *Embarrassingly Simple Unsupervised Aspect Extraction*. 2020. arXiv : 2004.13580 [cs.CL].

[TM08] Ivan Titov et Ryan McDonald. *Modeling Online Reviews with Multi-grain Topic Models*. 2008. arXiv : 0801.1063 [cs.IR].

[TS16] Zhiqiang Toh et Jian Su. "NLANGP at SemEval-2016 Task 5 : Improving Aspect Based Sentiment Analysis using Neural Network Features". In : *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. San Diego, California : Association for Computational Linguistics, juin 2016, p. 282-288. DOI : 10.18653/v1/S16-1045. URL : https://aclanthology.org/S16-1045.

[VPM20] Danny Suarez Vargas, Lucas R. C. Pessutto et Viviane Pereira Moreira. *Simple Unsupervised Similarity-Based Aspect Extraction*. 2020. arXiv : 2008.10820 [cs.CL].

[Xen+16] Dionysios Xenos et al. "AUEB-ABSA at SemEval-2016 Task 5 : Ensembles of Classifiers and Embeddings for Aspect Based Sentiment Analysis". In : *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. San Diego, California : Association for Computational Linguistics, juin 2016, p. 312-317. DOI : 10.18653/v1/S16-1050. URL : https://aclanthology.org/S16-1050.

[Yan+20] Zhilin Yang et al. *XLNet : Generalized Autoregressive Pretraining for Language Understanding*. 2020. arXiv : 1906.08237 [cs.CL].

[Zho+17] Lin Zhouhan et al. "A Structured Self-attentive Sentence Embedding". In : *International Conference on Learning Representations* (2017).

# Annexe

## Segmentation des sentiments

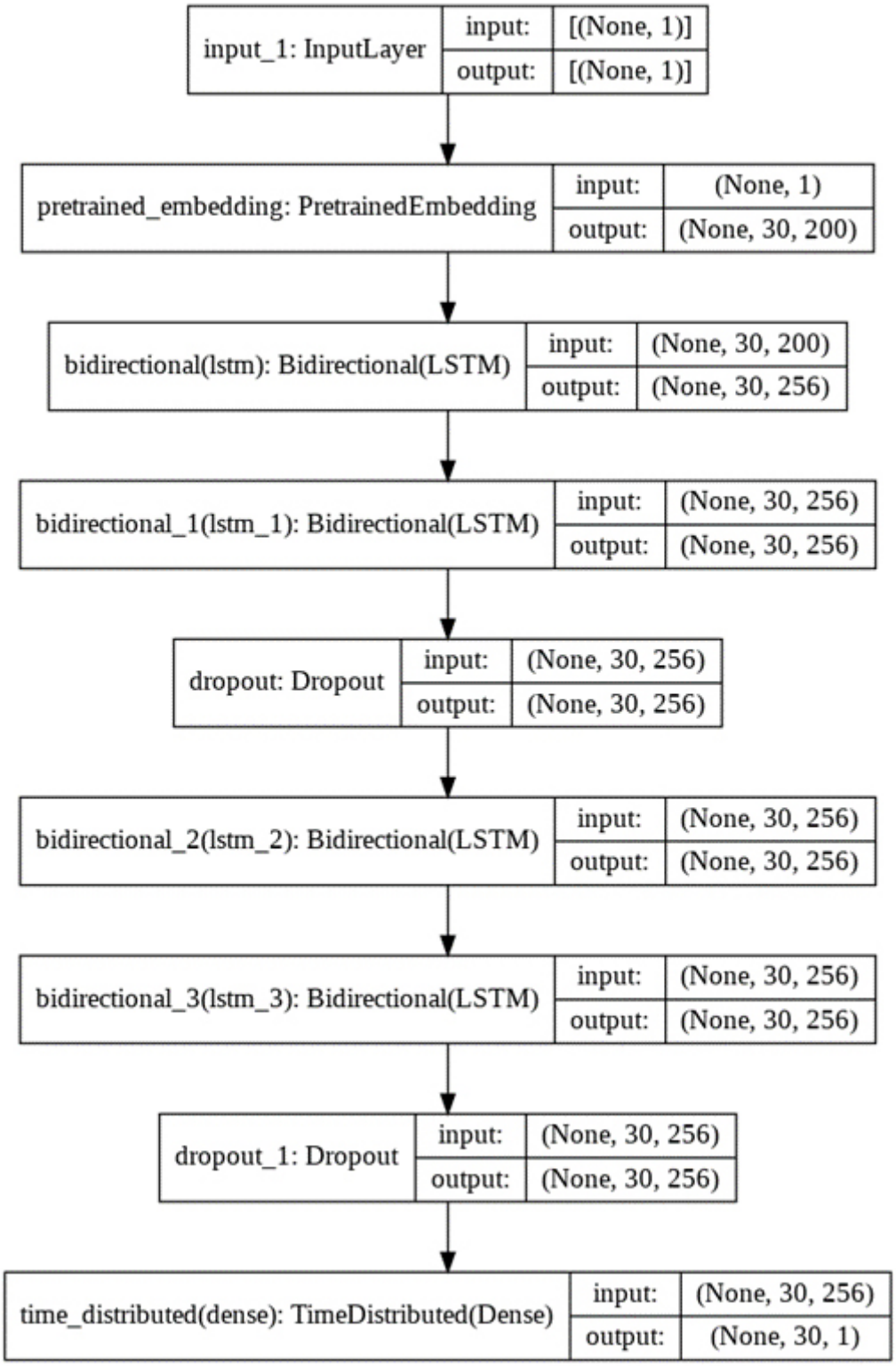


FIGURE 57 – Architecture BiLSTM utilisée dans la segmentation des sentiments
Le schéma a été généré avec Keras et contient des détails d'implémentation tels que la taille des paramètres.

### Pistes précédentes pour détecter les aspects

Initialement, l'algorithme pour repérer les phrases abordant un aspect se basait sur un auto-encodeur. L'idée était de pouvoir encoder une représentation de la phrase ou du segment en incluant le contexte dans lequel les termes ont été employés. Une telle représentation allait être compressée par l'auto-encodeur, entraîné avec cette représentation en entrée pour prédire cette même représentation en sortie. La partie encodeur allait compresser cette représentation dans une dimensionnalité plus faible, poussant le réseau de neurones à synthétiser la représentation afin de ne garder que les caractéristiques les plus pertinentes. Ainsi, l'encodeur serait capable de produire des représentations vectorielles plus adapté pour la détection d'aspects. Afin d'obtenir une séparation des phrases selon les aspects, il aurait été envisageable d'utiliser une méthode de réduction de dimension telle que la t-SNE afin de pouvoir visualiser les segments de textes dans un plan en deux dimensions. L'identification des groupes se ferait pour un algorithme de clustering sur les représentations en deux dimensions.

Cette méthode n'a pas suffi pour identifier les aspects ciblés. En effet, le résultat final contenait une multitude de clusters qui pouvait s'apparenter à des sous-aspects ; d'autres, renfermaient des segments de textes mélangeant certains d'aspects. De ce fait, les clusters n'étaient pas toujours interprétables. Le résultat final n'était pas suffisamment clair et pertinent. Par ailleurs, varier la représentation initiale n'a pas suffi et plusieurs pistes ont été tentées. En effet, changer la représentation vectorielle pour qu'elles soient générées par des modèles de langues plus robustes tel que BERT, varier l'algorithme de clustering ainsi que ces paramètres ou encore passer à une représentation par mot (une représentation constituée de l'embedding du mot et de la phrase au quelle il appartient) n'engendraient pas de résultats suffisamment satisfaisants et les mêmes problèmes revenaient.

Cela peut être dû au fait que la quantité de données n'était pas suffisante pour que cette compression par le réseau de neurones soit pertinente. L'idée d'utiliser une méthode basée sur des auto-encodeurs provient des techniques de topic modeling dites contextualisées qui compressent une représentation d'un texte combinée avec sa distribution des topics calculée au préalable par des méthodes plus rudimentaires (LDA par exemple).

De ce fait, il a paru nécessaire de changer de paradigme pour pallier ce problème de détection d'aspects avec les données à disposition.