Analyse d'opinion pour l'optimisation de la relation client : optimisation de services de gestion de communautés et d'outils de communication à destination du client.

Alexis Blandin

THÈSE DE DOCTORAT DE

L'UNIVERSITÉ BRETAGNE SUD

ÉCOLE DOCTORALE N° 644
Mathématiques et Sciences et Technologies de l'Information et de la Communication en Bretagne Océane
Spécialité : Informatique et Architectures Numériques

Par

Alexis BLANDIN

«Analyse d'Opinions pour l'Optimisation de la Relation Client »

« Optimisation de services de gestion de communautés, et d'outils de communication à destination du client »

Thèse présentée et soutenue à « Vannes », le « 27 mars 2023 »
Unité de recherche :
Thèse N° : 658

Rapporteurs avant soutenance :

Frédéric Béchet	Professeur d'informatique - Aix Marseille Université Laboratoire Informatique et Systèmes - LIS UMR7020
Vincent Labatut	Maître de conférence HDR, UFR Sciences, Technologies Santé Centre d'Enseignement et de Recherche en Informatique (CERI) UPR 4128 LIA Laboratoire Informatique d'Avignon

Composition du Jury :

Attention, en cas d'absence d'un des membres du Jury le jour de la soutenance, la composition du jury doit être revue pour s'assurer qu'elle est conforme et devra être répercutée sur la couverture de thèse

Président :	Emmanuel Morin	Fonction et établissement d'exercice *(à préciser après la soutenance)*
Examinateurs :	Ioanna Galleron	PR - Université Sorbonne Nouvelle
	Frédéric Béchet	Professeur d'informatique - Aix Marseille Université
	Vincent Labatut	MCF HDR, UFR Sciences, Technologies LIA
	Jeanne Villaneau	Membre du corps professoral - IRISA - UBS
	Farida Saïd	MCF - LMBA - UBS
Dir. de thèse :	Pierre-François Marteau	Professeur d'informatique - IRISA - UBS

Remerciements

On dit qu'une thèse est une épreuve solitaire, une sorte de traversée du désert ou de marathon, qui dure au minimum trois ans. Or en ce qui me concerne, ces trois années ont été remplies d'embûches, parmi lesquelles rien de moins qu'une pandémie mondiale, qui a poussé le monde entier à se confiner.

Malgré ce contexte, il serait faux de dire que j'ai franchi ces épreuves seul et par ma seule volonté. Beaucoup m'ont offert le soutien nécessaire à ce que je m'accroche pendant ces 3 années. En premier lieu mes encadrantes et mon directeur de thèse qui, bien que en distanciel ont su m'accompagner autant dans mes réussites que dans les moments les plus durs. De même, l'ensemble du corps professoral avec qui j'ai pu interagir, a su être bienveillant et disponible pour me permettre d'achever cette thèse en donnant le meilleur de moi-même.

Outre ce soutien du monde académique, je ne saurai jamais assez remercier ma famille et en premier lieu mes parents, qui ont su me soutenir et me conseiller à la hauteur de leurs moyens et même plus. Enfin, en écrivant ces mots, j'ai une pensée toute particulière pour ma grand-mère Jeanine qui, bien qu'elle ait vu le début de ma thèse, ne me verra jamais docteur. Mais au fond je sais qu'elle a toujours cru en moi et en ma capacité à finir cette thèse de la meilleure des manières.

J'espère avoir été à la hauteur de sa confiance, de sa fierté et de son soutien, ainsi qu'à celui de tous mes proches.

"Κἀνταῦθα ὦ ξένε, τὸ νικᾶν αὐτὸν αὑτὸν πασῶν νικῶν πρώτε τε καὶ ἀρίστε"
Πλάτων , Νόμοι (626.e 2 – 5)

Résumé en français

Cette thèse a pour sujet l'*"Analyse d'opinion pour l'optimisation de la relation client"*; elle a été réalisée dans le cadre d'une convention *CIFRE*, en partenariat avec la société *UNEEK*, qui propose des services de gestion de relation client.

Le sujet est vaste et recoupe de nombreux domaines, comme l'apprentissage profond, le traitement automatique du langage, l'analyse et la détection d'émotions ou le marketing et les sciences sociales. Nos travaux abordent essentiellement deux problématiques : l'interrogation en français de données tabulaires et l'étude de newsletters.

Dans une première partie, nous explorons la possibilité de concevoir une interface de requêtes permettant d'interroger en langage naturel les bases de données tabulaires de l'entreprise.

Ensuite, nous expliquons comment et pourquoi nous avons recentré nos travaux sur l'étude d'un canal de communication particulier : les newsletters.

Nous étudions d'abord comment les émotions transmises par le texte des newsletters, influent sur leur perception, et comment les résultats de cette étude peuvent aider à leur rédaction et leur édition.

Ensuite, nous proposons une modélisation des newsletters sous forme de graphes hétérogènes, permettant de prendre en compte les aspects visuels des zones de textes et leur disposition dans la newsletter, en plus de leur contenu. Nous utilisons des techniques d'apprentissage profond telles que les réseaux de convolution de graphe, et les techniques d'attention pour prédire la performance des newsletters.

Cette modélisation originale des newsletters a produit des résultats encourageants pour la tâche de prédiction considérée. L'approche pourrait être approfondie dans de futurs travaux pour prendre en compte d'autres composantes significatives des newsletters, en particulier les images.

Contributions

[1, 2, 3, 4]

BIBLIOGRAPHIE

[1] A. Blandin, "Adaptation de ressources en langue anglaise pour interroger des données tabulaires en français (adaptation of resources in english to query french tabular data)." in Actes de la 28e Conférence sur le Traitement Automatique des Langues Naturelles. Volume 2 : 23e REncontres jeunes Chercheurs en Informatique pour le TAL (RECITAL), 2021, pp. 47–54.

[2] A. Blandin, F. Saïd, J. Villaneau, and P.-F. Marteau, "Automatic emotions analysis for french email campaigns optimization." in CENTRIC 2021, 2021.

[3] ——, "Dafnege : Dataset of french newsletters with graph representation and embedding." in International Conference on Text, Speech, and Dialogue. Springer, 2022, pp. 16–27.

[4] ——, "Graphical document representation for french newsletters analysis," in Proceedings of the 22nd ACM Symposium on Document Engineering, 2022, pp. 1–8.

Table des Matières

Liste des acronymes **11**

Liste des figures **15**

Liste des tables **17**

Introduction **19**

1 Travailler sur des données tabulaires en français **23**
1.1 Contexte, objectifs et données à disposition 23
1.2 L'interrogation de données structurées 24
1.2.1 Le Text2SQL 24
1.2.2 Autres méthodes de traitement de requête en langage naturel 25
1.3 Présentation des jeux de données 28
1.4 Présentation de l'architecture des transformers 31
1.4.1 Le mécanisme d'attention 31
1.4.2 Les Transformers 32
Multi-Head Attention 33
1.5 Adaptation d'un modèle anglophone à un modèle francophone, l'exemple des données tabulaires et du modèle TAPAS 34
1.5.1 Traduction des données 34
1.5.2 Présentation du modèle 35
1.6 Adaptation du modèle pour les données francophones, et expérimentation 38
1.6.1 Méthode pour adapter le modèle sur une tâche donnée (fine tuning) . . . 38
Apprentissage du modèle 38
1.6.2 Expérimentation sur les données francophones 39
1.7 Conclusion et perspectives 41
Bibliography 41

2 La communication client par mail **47**
2.1 Problématiques qualitatives et quantitatives des données. 47
2.2 La communication client 47
2.3 La communication par mail 49

2.4 Établissement d'un jeu de données de newsletters 50
2.4.1 Caractéristiques du mail 52
2.4.2 Caractéristiques du receveur 54
2.4.3 Caractéristiques de l'envoyeur 55
2.4.4 Réponses aux mails 55
2.4.5 Forme finale des données 56
2.5 Conclusion 56
Bibliography 57

3 Analyse d'Opinion et d'Émotion 61
3.1 L'analyse d'opinion et d'émotion, une problématique large 61
3.1.1 L'analyse d'opinion 62
3.2 Les différents modèles d'émotions dans le texte 64
3.2.1 Modèles catégoriels 64
Le modèle d'Ekman 65
Le modèle Narasava 65
3.2.2 Modèles dimensionnels 67
Le modèle circomplexe 67
Le modèle PAD *Pleasure-Arousal-Valence* 68
Le modèle de Plutchnik 69
3.2.3 Bilan des modèles 70
3.3 Construction d'un modèle d'analyse d'émotions 71
3.3.1 Approche par lexique 73
Extraction de caractéristiques 74
Résultats statistiques 74
3.3.2 Classification non supervisée 76
Approche par k-moyennes 77
3.3.3 Classes de performance 78
3.3.4 Classification supervisée 79
3.3.5 Approche par modèle entraîné à détecter les émotions 82
Comparaison des méthodes de prédiction de performance des newsletters 84
3.4 Conclusions et perspectives 87
Bibliography 88

4 Modélisation des newsletters 93
4.1 Retour sur la modélisation du contenu de newsletters 93
4.1.1 Motivations pour une nouvelle représentation 93
4.1.2 Mise en oeuvre 94

Détection des zones de texte 94
Alignement des zones de texte avec les divisions *HTML* 96
Construction du graphe 99
4.1.3 Ingénierie des caractéristiques (*feature engineering*) 102
Features textuelles 102
Features graphiques 103
4.2 Convolutions de Graphes 104
4.2.1 Présentation du champ de recherche 104
4.2.2 Catégorisations des GNN (*Graph Neural Networks*) 105
4.2.3 Les approches spatiales 107
4.2.4 Les approches spectrales 107
4.2.5 Le modèle R-GCN 108
4.3 Résultats 110
4.3.1 Influence de la Convolution 111
4.3.2 Influence de l'Attention 113
Avantages du readout d'attention sur l'interprétabilité des prédictions . . 115
4.3.3 Influence du dropout 116
4.3.4 Influence des features 119
4.4 Conclusions et perspectives 120
Bibliography 121

Conclusion **125**

Liste des acronymes

NLP	Natural Language Processing
TAL	Traitement Automatique du Langage
CRM	Customer Relationship Management
GCN	Graphical Convolutional Network
ROI	Return On Investment

TABLE DES FIGURES

1 Schéma représentant l'organisation des différents acteurs (en bleu), des problématiques de communications associées (en rouge) et les échanges de données (en vert). 20

1.1 Framework d'un système Text2SQL 25
1.2 Schéma proposant une topologie des différentes méthodes de Text2SQL 26
1.3 Diagramme représentant la distribution des requêtes en français selon leur longueur 30
1.4 Diagramme représentant la proportion de chaque agrégateur SQL dans les différentes partitions du jeu de données 30
1.5 Schéma représentant comment l'attention se porte de manière différenciée selon les mots 32
1.6 Schéma représentant comment est calculée la *Scale Dot-Product Attention* 33
1.7 Schéma de principe de l'attention multi-têtes (*Multi-Head Attention*) 34
1.8 Schéma représentant l'architecture de *Transformers* dans son ensemble 35
1.9 Schéma représentant la procédure de pré-apprentissage (pre-training) et d'apprentissage fin (fine-tuning) de BERT. Les mêmes architectures sont utilisées dans les deux cas, cependant le pre-training se focalise sur quelques paramètres, qui sont par la suite réutilisés pour le fine-tuning, ce dernier entraînant tous les paramètres suivant la tâche demandée. 36
1.10 Schéma représentant l'architecture globale du modèle TAPAS 37
1.11 Modèle TAPAS (en bas) appliqué sur la question "Total number of days for top two", prédiction de la cellule (à droite), donnée pour les colonnes selectionnées (en gras), et prédiction de l'aggrégateur (à gauche). 38
1.12 Schéma représentant le processus d'obtention des modèles 40

2.1 Modèle conceptuel de la réponse de l'internaute face à la réception d'un email commercial 51
2.2 Exemple de Newsletter envoyé par UNEEK, via son outil d'édition 52
2.3 Schéma décrivant le processus de l'*A/B testing* 53
2.4 Répartition des newsletters selon les clients de UNEEK les ayant envoyées. . . . 54
2.5 Modèle d'intention d'usage intégrant le plaisir perçu 55

3.1 Représentation graphique de l'opinion des mots suivant les axes de polarité et subjectivité 63

3.2 Trois tâches de l'ABSA à travers une phrase d'exemple venant du jeu de données du SemEval ABSA dataset 2016. La phrase a deux cibles d'opinions, "sushi" et service. La catégorie "sushi" de sushi est "Food" avec pour attribut "Quality" et pour polarité "Positive". Pour le "service" on y associe la catégorie "Service" et une opinion elle aussi "Positive" 64

3.3 Visages représentant les faciès présentant les 6 émotions d'Ekman utilisées par Paul Ekman pour définir les émotions de bases. De gauche à droite et de haut en bas : la joie, la surprise, la peur, la colère, le dégoût et la tristesse 66

3.4 Modèle circomplexe de Russel 67

3.5 Concepts d'émotions au sein du modèle tri-dimensionnel de Latinjak 69

3.6 Schéma de la roue des émotions de Plutchnik 70

3.7 Visualisation en 3D du modèle de Plutchnik 71

3.8 Visualisation en 3D du modèle du sablier d'émotions 72

3.9 Arborescence des principales méthodes de détection d'émotions dans le texte . . 73

3.10 Schéma présentant le processus de détection d'émotions par sac de mots 74

3.11 Projection via t-SNE de notre jeu de données 76

3.12 Diagramme en boîtes à moustaches comparant les taux de cliques pour chacun des deux clusters obtenus par k-moyennes, avec 8 composantes principales. Le test de Student nous indique que ces deux clusters ne sont pas statistiquement différenciables par le taux de clique. Ce test fut effectué sur un jeu de donnée de taille N=973, comprenant les données filtrées par la suite (doublons, etc). Les conclusions sur le jeu de 799 newsletters sont les mêmes 79

3.13 Distribution du score de silhouette pour les deux clusters suivant le taux de cliques bas (en rouge) ou élevé (en vert). 80

3.14 Distribution du score de silhouette pour les deux clusters suivant le taux de cliques bas (en rouge) ou élevé (en vert) ; mais sans les features issues de l'analyse de l'objet du mail. 80

3.15 Schéma représentant la méthode d'apprentissage transfert, visant à adapter le modèle T5 de détection d'émotion depuis l'anglais vers le français. 83

3.16 Matrice de confusion des prédictions du modèle T5 ré-entraînés pour la détection d'émotions en français. 85

3.17 Exemple de phrase traitée par les deux approches. Ici 0^{em} désigne le vecteur nul pour l'espace d'émotions considérées. On peut voir que l'approche par sacs de mots ne permet pas de labéliser correctement les émotions du fait de l'absence de marqueur lexical d'émotion. En revanche le modèle T5 adapté au français y parvient. 85

4.1 Schéma représentant la segmentation proposée par LayoutParser utilisant *Detectron2*. Image tirée du dépôt github de LayoutParser (https://github.com/Layout-Parser/layout-parser) 95

4.2 Illustration de l'algorithme de correction de la segmentation dans le cas où deux zones détectées se chevauchent 96

4.3 Illustration de l'apport de l'algorithme "Levenshtein coulissant" par rapport à une distance de Levenshtein simple. Notre exemple prend le cas d'une phrase, mais en pratique nous l'appliquons surtout sur des paragraphes. 97

4.4 Diagramme de classes représentant les objets "box" et "div" référençant respectivement les zones de texte détectées et non détectées 98

4.5 Schéma représentant l'alignement entre les textes issus de l'HTML et les zones de texte détectées dans l'image de la newsletter. Dans le cas où on ne peut pas associer de zone graphique au texte, nous ne considérons que le texte. 99

4.6 Condition de connectivité entre deux zones de texte 100

4.7 Condition de visibilité telle que décrite dans par Locteau et al. 100

4.8 Exemple de newsletter, avec en bleu les zones de textes détectées par l'OCR, et en jaune, celles qui ne le sont pas 101

4.9 Visualisation du graphe modélisant la newsletter donnée en 4.8. Les noeuds bleus représentent les zones détectées, et les noeuds jaunes celles qui ne le sont pas . . 102

4.10 A gauche, la représentation d'une convolution sur une image vue comme "graphe grille", à droite le même concept appliqué à un graphe quelconque 105

4.11 Arbre regroupant certaines architectures suivant leurs catégories, ainsi que les champs d'applications possibles 107

4.12 Architecture de notre modèle visant à classifier les newsletters représentées en graphes 110

4.13 Courbes d'apprentissage moyen pour chaque nombre de couches de convolutions. 112

4.14 Courbes d'apprentissage moyennes pour 6 couches de R-GCN mais avec différentes fonctions readout. 114

4.15 Courbes d'apprentissage moyennes pour 3 couches de convolution, avec différentes fonctions de readout. 115

4.16 Graphe représentant le même exemple de newsletter que les figures 4.8 et 4.9, mais les noeuds sont ici colorés suivant les poids d'attention assignés à chaque noeud lors du readout. 116
4.17 Illustration du mécanisme de dropout 117
4.18 Courbes d'apprentissage moyennes pour 3 couches de convolution utilisant le readout d'attention, mais avec un taux de dropout variable. Les moyennes sont réalisées selon 5 jeux d'apprentissages, de validation et de test différents. Ici les courbes s'arrêtent à 150 époques, le sur-apprentissage rendant le graphe illisible au-delà. La courbe grise représente la courbe d'apprentissage du modèle n'utilisant pas le readout d'attention. 118

Liste des tableaux

1.1 Exemple d'une question du jeu de données WikiSQL, ainsi que les entêtes de la table associée, et la requête SQL correspondante 29
1.2 Exemple du résultat d'une traduction d'un item du jeu de données WikiSQL . . 29
1.3 Exactitude (accuracy) sur le jeu de test et développement des deux modèles anglophone et francophone. 40

3.1 Corrélation de Pearson entre les features et les indicateurs de performance. . . . 75
3.2 Nombre de clusters avec le meilleur score de silhouette, selon le nombre de composantes principales gardées. 78
3.3 F1-score, précision et rappel, de la tâche de classification suivant les modèles, et avec ou sans les features issues de l'objet du mail 81
3.4 Résultats de la classification avec seulement une feature, ou bien toutes sauf celle testée. 82
3.5 Résultat du modèle T5 à l'issue d'un ré-apprentissage (*fine tuning*) visant à prédire des émotions sur des données traduites en français, et la différence de performance par rapport au modèle originel en anglais. 84
3.6 Corrélation de Pearson entre les caractéristiques des *newsletters* et leurs performances, suivant les deux approches : par lexique (en noir) et avec le modèle T5 réappris (en bleu) 86
3.7 F1-Score des différents classifieurs exploitant les différents plongements. 87

4.1 Mesure selon plusieurs métriques de la performance de prédiction de la classe de performance en faisant varier le nombre de couches de convolution, avec 12 jeux d'apprentissage/validation/test différents. Le modèle avec 0 couche de convolution est considéré comme une baseline. 112
4.2 Écart type de chacune des métriques d'évaluation présentée en table 4.1 113
4.3 Performances des modèles avec un readout attention, suivant différent taux de dropout. Ces résultats sont les moyennes de métriques calculées sur 5 modèles de même architecture entraînés selon différentes séparations du jeu de données 4.18, 4.15, et 4.14. 119

4.4 Résultats de modèles ayant la même architecture (trois couches de convolution, readout simple), entraînés selon de jeux de features différents. Ces résultats sont les moyennes de métriques réalisée sur 5 modèles selon différentes séparations du jeu de données. Ces séparations sont les mêmes que pour les figures 4.18, 4.15, 4.14, et le tableau 4.3 120

INTRODUCTION

Cette thèse porte sur l'*"Analyse d'opinion pour l'optimisation de la relation client"*, un sujet très large qui recouvre des problématiques issues de plusieurs champs de recherche. Régie par une convention CIFRE, elle se place dans le contexte particulier de l'entreprise *UNEEK* qui propose à ses clients des outils de gestion de la relation client.

L'analyse ou la fouille d'opinion en tant que vecteur d'optimisation de la relation client est un sous-domaine de l'intelligence artificielle. Les entreprises intègrent ainsi de plus en plus dans leur outillage de gestion de la relation client des fonctions d'analyse automatisée des commentaires et courriers pour en extraire plusieurs informations comme des opinions portant sur l'entreprise, des opinions sur les produits ou les services proposés, ou encore pour ajouter des fonctions de génération automatisée de réponses aux messages, en accord avec le contexte.

Les sources d'information exploitables en temps réel engendrent une volumétrie des données de plus en plus importante induisant un contexte "big data" qui sature de fait les capacités de traitement manuel des entreprises. La nécessité d'une automatisation des processus de traitement des commentaires des clients, des utilisateurs et des usagers devient ainsi extrêmement prégnante.

Parallèlement, les progrès récents en traitement automatique du langage naturel (TAL, en anglais NLP) qui découlent principalement des approches d'apprentissage machine et d'apprentissage profond, offrent de nouvelles perspectives d'applications très prometteuses. Ainsi, des architectures fondées sur des plongements de mots (Word2vec, fastText, Transformer) ont ouvert la voie à un foisonnement de travaux ciblant des tâches réputées très complexes au cœur de l'intelligence humaine et artificielle comme la compréhension du langage naturel qui présuppose le raisonnement et l'inférence.

L'IA (Intelligence Artificielle) conversationnelle, en première ligne dans la constitution des interfaces utilisateur naturelles, est devenue un domaine en pleine croissance qui attire de plus en plus les chercheurs et les acteurs économiques issus du traitement du langage naturel (NLP), de la recherche d'information (IR) et de la communauté de l'apprentissage automatique (machine learning).

Dans le contexte de l'entreprise UNEEK, les travaux présentés visent à offrir des pistes pour améliorer la relation client et donc la communication via des solutions d'intelligence artificielle, que ce soit par le biais d'agents conversationnels ou d'autres canaux, afin de proposer ces solutions aux clients de l'entreprise. Plus précisément, ils combinent : apprentissage profond, détection d'émotions et d'opinions et utilisation de graphes pour la modélisation des données.

UNEEK est une entreprise proposant des solutions pour la gestion de relation client (CRM).

Ainsi, nous désignons ici par « clients » les clients de *UNEEK* tandis que nous appellerons « contacts » les personnes utilisant le CRM (*Customer Relationship Management*. L'optimisation de cette relation client est un enjeu stratégique pour *UNEEK*. Cependant, cette optimisation ne peut se faire qu'à travers un medium en particulier, surtout que dans notre cas, nous nous intéressons à la communication client d'un client de UNEEK (entreprise ou structure) vers un contact ou un client propre, ce qui est différent de la plupart des études sur la relation client qui étudie le *feedback* renvoyé par le client final. La figure 1, représente le framework d'organisation entre les différents acteurs ainsi que les problématiques correspondant à chaque échelon d'échange.

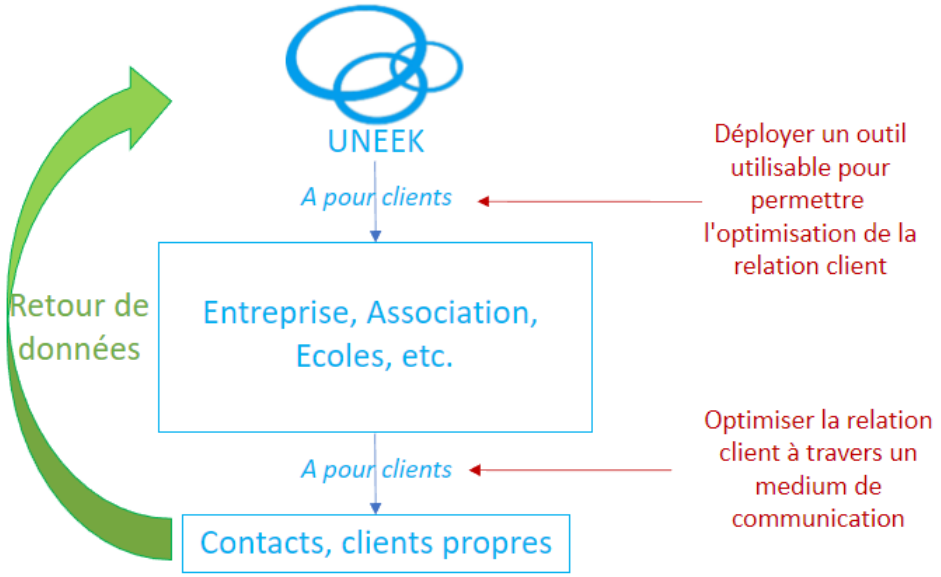


FIGURE 1 : Schéma représentant l'organisation des différents acteurs (en bleu), des problématiques de communications associées (en rouge) et les échanges de données (en vert).

Dans un premier temps, nous avons donc envisagé un chatbot qui interagirait avec les contacts, qui par la suite aurait vocation à être optimisé via l'analyse de l'opinion renvoyée par le contact. Ainsi, dans un premier chapitre, nous verrons comment nous avons abordé le contexte de données propre à l'entreprise à travers l'étude de requêtes en langage naturel de données tabulaires. En effet, dans un but de garantir la meilleure relation client possible et dans une volonté de travailler sur du langage naturel, il nous faut en amont permettre de requêter les données de l'entreprise via un chatbot performant. C'est dans cette perspective que nous explicitons les différents aspects techniques liés à l'apprentissage profond que nous avons utilisés par la suite, que ce soit les modèles d'attention, ou les problématiques liées à l'adaptation de modèles anglophones en français.

Cependant, au fil de notre étude, il est apparu que compte tenu du contexte UNEEK, cette création de chatbot n'était pas réalisable dans le cadre d'une thèse de doctorat, de part notamment le manque de données labélisées. C'est pourquoi nous avons réorienté notre travail sur l'étude d'un autre moyen de communication client : les échanges de mail et plus particulièrement, l'étude de ceux envoyés par les clients à leurs contacts. *UNEEK* propose en effet à ses clients un moyen d'éditer et d'envoyer des newsletters à leurs listes de contacts. En effet, offrir un moyen d'optimiser cette communication par mail à toute une communauté est au cœur des activités de *UNEEK*.

Dans un deuxième chapitre, nous expliquons comment et pourquoi nous avons recentré notre étude sur les newsletters, un cas particulier de relation et de communication.

Beaucoup d'études de la relation entre une entreprise (ou ici un client de *UNEEK*), et un client (ici un contact), se focalisent sur l'analyse du retour du contact ou « feedback ». Or, nous avons préféré prendre le point du vue inverse, en l'occurrence chercher à analyser et optimiser la communication venant du client du CRM, vers sa liste de contacts. Ce changement de perspective nous est possible grâce aux données mis à disposition par l'entreprise, qui nous ont permis de constituer un jeu de données de « templates » de newsletters.

Nous avons donc cherché à déterminer dans quelle mesure le ton employé dans une newsletter éditée par un client du CRM (déterminé par l'analyse d'émotions et d'opinion), influe sur la perception et la performance d'une newsletter. Le but étant de pouvoir proposer une indication qualitative à l'éditeur de newsletters en prédisant l'impact de sa newsletter avant son envoi, pour qu'il puisse l'optimiser.

Enfin, les troisième et quatrième chapitres cherchent à déterminer respectivement comment l'analyse d'opinions et d'émotions d'une part, et une modélisation sous forme de graphes du texte et de sa mise en page d'autre part, peuvent influer sur les performances d'une newsletter. Nous proposons donc ici une modélisation originale de nos données, via des graphes, et des convolutions de graphes, qui permet de rajouter un contexte visuel au texte des newsletters et de modéliser la mise en page du texte.

Du fait des problématiques et des approches diverses abordées, nos travaux sont essentiellement exploratoires. Cependant, les résultats obtenus sur ces idées assez novatrices sont encourageants, et pourraient être approfondis dans des travaux futurs. Quelques pistes en ce sens sont données dans la conclusion.

Chapitre 1

Travailler sur des données tabulaires en français

1.1 Contexte, objectifs et données à disposition

Cette thèse s'est déroulée dans le cadre d'un partenariat CIFRE avec l'entreprise *UNEEK*. *UNEEK* propose à ses clients une plateforme offrant plusieurs outils, et notamment un outil de gestion de relation client ou *CRM*. Le sujet de la thèse, qui porte sur l'analyse d'opinion en vue d'optimiser cette relation client, a naturellement conduit à s'intéresser à ce CRM.

UNEEK propose en effet un service de gestion de relation client qui repose sur l'envoi de formulaires, dont les informations sont par la suite stockées dans des tables SQL sur lesquelles des requêtes sont possibles. Chaque client de *UNEEK* possède ainsi son propre environnement CRM composé de tables paramétrées selon ses besoins, ainsi que les formulaires associés. Cependant les usagers peuvent trouver l'outil proposé austère ou peu pratique pour l'extraction d'informations dans ces tables, d'où l'idée de passer par un traitement du langage naturel pour requêter sur les données du CRM.

Les avancées en traitement automatique du langage permettent, depuis quelques années, d'envisager des applications qui offrent des services à plus forte valeur ajoutée, et dans des milieux professionnels très diversifiés. Ainsi l'interrogation en langage naturel de données issues d'un CRM constitue un exemple marquant de services innovants mis en ligne ces dernières années pour enrichir les échanges entre clients et fournisseurs. Notre première idée pour améliorer la relation client chez *UNEEK*, a été de concevoir un agent conversationnel (ou chatbot) qui puisse répondre à des requêtes en langage naturel sur la base de données du CRM.

Avant de proposer une optimisation de la communication client via un outil de requêtage, il convient de qualifier l'outil lui-même. Cet outil doit pouvoir à partir d'une requête formulée en langage naturel, déduire une information contenue dans le CRM. Pour garantir la bonne qualité de la relation client, il convient donc avant tout de s'assurer de la fiabilité des réponses produites. C'est pourquoi nous nous sommes tout d'abord orientés sur l'étude de l'interrogation en langage naturel de données structurées.

L'utilisation du langage naturel pour interroger les bases de données (NLIDB[1]), est devenu un enjeu central pour les entreprises qui désirent simplifier la communication avec leurs contacts, et par la suite l'optimiser.

1.2 L'interrogation de données structurées

L'interrogation de bases de données structurées par des requêtes en langage naturel est une problématique très large et étudiée depuis longtemps. Cette thématique des interfaces en langage naturel (NLIDB[2]), est devenue centrale pour les entreprises qui veulent établir une communication plus naturelle avec leurs contacts. La tâche est ancienne [1, 2] et encore dynamique aujourd'hui [3].

1.2.1 Le Text2SQL

On peut noter deux catégories de méthodes pour aborder la problématique des requêtes en langage naturel (LN). La première, la plus ancienne, consiste en la traduction de la requête en LN en une requête en langage formel. S'il existe plusieurs langages formels de requêtes (SQL, SPARQL, ...), nous détaillons ci-dessous la traduction de langage naturel en requête SQL ou : Text2SQL.

Le Text2SQL est un domaine de recherche en traitement automatique du langage naturel (TAL), qui vise à convertir un texte LN en une requête SQL qui peut être exécutée sur une base de données. Un schéma présentant le framework d'un système text2SQL est donné en figure 1.1 Ce domaine de recherche est ancien et encore actif de nos jours et le sujet a fait l'objet d'une thèse complète soutenue en 2022 [4]. C'est pourquoi nous ne détaillerons pas un état de l'art complet mais quelques méthodes saillantes, afin de mieux situer nos travaux. De plus, des états de l'art complets rendent déjà assez bien compte de cette problématique de recherche [5, 6, 7, 8, 9, 10]. Ici nous nous basons sur l'étude récente de Deng et al.[5] pour proposer une vision globale de ce champ de recherche.

Parmi les méthodes développées pour améliorer ces modèles de traduction de requête naturel en SQL, les premières méthodes que l'on peut citer sont celles fondées sur des règles [11, 12]. Ces approches relativement simples peuvent convenir pour un usage sur des requêtes simples ou à destination de bases de données simples. Toutefois des paramétrages plus complexes [13] ont permis une meilleure généralisation des modèles.

Au sein des autres méthodes pour améliorer ces modèles de Text2SQL [5], on peut également noter les travaux qui cherchent à annoter les données que ce soit pour une tache faiblement supervisée [14] ou complètement supervisée [15, 16].

1. Natural Langage Interface Database
2. Natural Langage Interface for Database

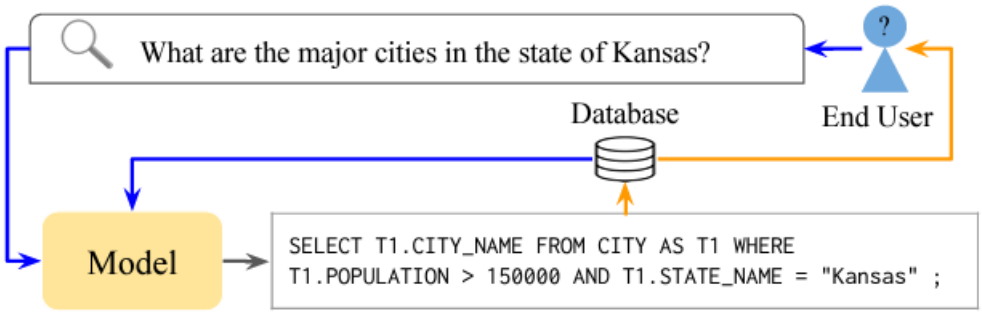


FIGURE 1.1 : Framework d'un système Text2SQL[5]

Un autre moyen d'augmenter les performances des modèles [5] est l'enrichissement de données. Cet enrichissement permet de gérer des questions plus complexes ou moins courantes [17, 18] notamment en paraphrasant [19, 20] la requête, et en remplissant plusieurs schémas prédéfinis pour enrichir les données et ainsi avoir un éventail plus large de requêtes.

Enfin, on peut citer les méthodes d'encodage et de décodage pour traiter les données [5], méthodes que nous ne détaillerons pas ici mais qui ont fourni de nombreuses avancées.

Une vision d'ensemble des méthodes évoquées ici est donnée en figure 1.2 [5].

Ainsi la tâche de text2SQL s'est beaucoup développée ces dernières années, avec des innovations majeures apportées par la dynamique "apprentissage profond" telles que les les approches à base de LSTM (Long-Short-Term-Memory) [21], la *self-attention* [22], etc. Le but des recherches dans cette tâche est notamment d'arriver à un système assez robuste pour requêter sur plusieurs domaines et être le plus "générique" possible, c'est à dire sans être liée à un thème ou une structure syntaxique en particulier.

Cependant, l'une des limites de l'application des modèles basés sur l'apprentissage est la nécessité de disposer un grand nombre de données généralement annotées correspondant au domaine étudié. Or dans notre cas très spécifique et avec des requêtes en français, établir un tel jeu de données nous a semblé hors de notre portée. On peut toutefois mentionner les travaux de B.Couderc et J.Ferrero [23] qui ont appliqué certaines de ces méthodes à l'interprétation de requêtes en français. Toutefois, ces difficultés quant au besoin de données nous ont poussé à nous tourner vers un autre moyen de traiter nos requêtes en langage naturel.

1.2.2 Autres méthodes de traitement de requête en langage naturel

Le vif intérêt suscité par les problématiques liées à l'interprétation des requêtes en langage naturel, est visible à travers l'apparition de jeux de données tels que SQuAD [24], associé au benchmark GLUE [25]. Cette tâche consiste en l'interprétation d'une requête en langage naturel

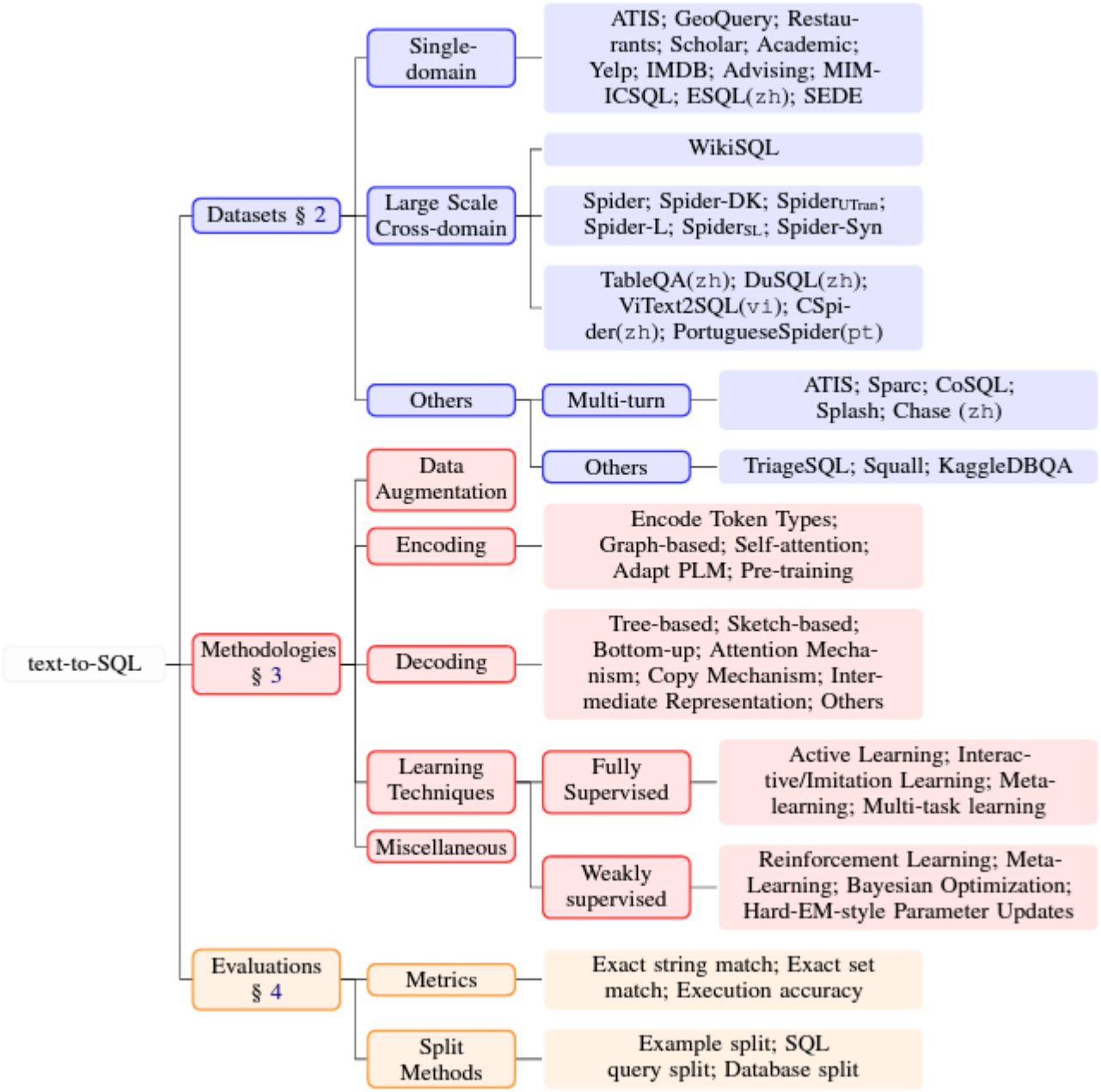


FIGURE 1.2 : Schéma proposant une topologie des différentes méthodes de Text2SQL[5]

sur un texte, à laquelle le système répond en produisant un extrait du texte contenant la réponse à la question posée.

Dans l'entreprise *UNEEK*, les données du CRM sont stockées sous forme tabulaire. Si dans SQuAD le contexte est représenté par un texte, il s'agit ici d'une table SQL. Ainsi on cherchera plutôt à déterminer à partir d'une requête en langage naturel une requête SQL.

Cette tâche est connue et documentée depuis longtemps en TAL, et consiste essentiellement en un parsing sémantique de la requête en langage naturel, où la question est transformée en une forme logique (ici, une requête SQL), pour interroger la table [26, 27].

Cependant une autre méthode de traitement de la langue phrase-à-phrase a été proposée (*Sequence to sequence, ou Seq2Seq*) [28, 29]. Cette approche a permis d'envisager d'autres méthodes pour la traduction en requête SQL, qui ont influencé l'état de l'art. [14].

Enfin, dans un contexte de conversation et d'un dialogue composé d'interactions successives, comme cela peut être le cas dans l'usage d'un chatbot, on peut imaginer un modèle qui vise à *rechercher* l'information, et apprend le meilleur moyen de la retrouver. Cette approche est celle proposée par M.Iyyer et al. [30], et semble particulièrement performante pour les requêtes complexes qui, une fois décomposées en plusieurs formes simples, offrent un *cheminement logique* à retrouver par le modèle.

Cependant, toutes ces méthodes visent à générer, à partir d'une requête en langage naturel, une expression logique la plus formelle possible. Cette approche pose plusieurs problèmes.

Tout d'abord, elle nécessite un gros travail de labélisation des données. Or, outre le risque introduire un biais de labélisation [31, 32], la constitution d'un tel jeu de données de requêtes labélisées nous a paru hors de portée durant cette thèse. Ainsi, nous avons fait le choix nous tourner vers des méthodes qui ne cherchent pas à générer une forme logique, mais à raisonner directement sur les tables [33, 34, 35].

Plusieurs des modèles présentés précédemment [27, 30, 14], sont fournis avec un jeu de données pour résoudre des tâches de QA (*Question Answering*). On peut alors remarquer que, sur ces données, le modèle TAPAS développé par [36], donne actuellement les meilleurs résultats. De plus, comparé aux autres architectures raisonnant sur les tables [33, 34, 35], TAPAS possède une architecture plus simple avec un unique encodeur, et semble plus souple quant aux types de questions allouées (contexte conversationnel, questions avec agrégation, etc). De plus, l'architecture de TAPAS permet de séparer son apprentissage en deux étapes : le *pre-training* [37] et le *fine-tuning*, ce qui va nous être utile par la suite. Ainsi, nous nous sommes tournés vers une architecture spécifique qui reprend ce concept de raisonnement à partir des tables directement : TAPAS , qui utilise d'une manière originale l'architecture BERT [38].

En effet, la grande majorité des ressources sont conçues pour un usage en langue anglaise. On peut donc se demander dans quelle mesure une nouvelle collecte de données est nécessaire pour obtenir un modèle équivalent à TAPAS pour le français. C'est pourquoi nous proposons une traduction de l'anglais vers le français du jeu de données proposé par [14], ainsi que son évaluation sur une version ré-entraînée du modèle TAPAS. L'objectif est de déterminer si ce

jeu de données obtenu par traduction d'une ressource anglophone est suffisant pour obtenir un modèle de traitement de questions en langage naturel sur des données tabulaires en français. Plus précisément, nous souhaitons établir dans quelle mesure ce modèle ré-appris sur les données traduites produit des résultats comparables à ceux obtenus pour l'anglais sur une tâche équivalente.

1.3 Présentation des jeux de données

Les ressources actuellement disponibles consistent en plusieurs jeux de données, réalisés pour répondre à la tâche d'analyse de données tabulaires, soit en se focalisant sur des questions générales et complexes [27], plusieurs questions simples [30], soit en cherchant une traduction la plus fidèle qui soit d'un langage naturel vers le SQL [14].

- **WIKITableQuestion** [27] est un jeu de données composé de questions sur des tables HTML issues de Wikipedia auxquelles sont associées des questions complexes réalisées par des humains à qui il a été demandé de créer, suivant une table donnée, des questions complexes dont la réponse nécessite plusieurs opérations sur la table (agrégation, comparaisons, superlatifs (minima et maxima), opérations mathématiques). Au total le jeu de données comprend 22 033 questions sur 2 108 tables.

- **SQA** [30] : Ce jeu de données a été construit en demandant à des humains de décomposer un sous-ensemble de questions hautement compositionnelles de WIKITQ, de sorte que chaque question décomposée résultante puisse être renseignée par une ou plusieurs cellules d'une table SQL. L'ensemble final se compose de 6 066 séquences de questions avec 2,9 questions par séquence en moyenne.

- **WikiSQL** [14] : ce jeu de données se concentre sur la traduction de texte en expression SQL. Il a été construit en demandant à des humains de paraphraser une question basée sur un modèle en langage naturel, deux autres personnes étant invités à vérifier la qualité des paraphrases proposées. Le résultat est un ensemble de 80 654 questions sur 24 241 tables issues de Wikipédia.

Entre ces trois jeux de données, notre choix s'est porté sur WikiSQL, car c'est le plus important d'un point de vue de la volumétrie, mais aussi parce qu'il fait office de benchmark pour cette tâche ([39], [40]). De plus dans leur article présentant le modèle TAPAS, [36] ont pu tester l'apprentissage par transfert de WIKISQL vers un autre jeu de données avec un certain succès. Notre expérience étant fondée sur cette application de l'apprentissage par transfert sur des données traduites, le choix de ce jeu de données semble justifié.

De la même manière que [41] ont proposé une traduction du jeu de données SQuAD en utilisant l'API de google traduction, nous proposons une version traduite du jeu de données WikiSQL en utilisant cette même API. Cette tâche de traduction comporte trois étapes :

- la traduction de la question en langage naturel de l'anglais vers le français ;
- la traduction des entêtes des colonnes de la table lorsque cela est nécessaire ;
- le remplacement des entêtes dans les requêtes SQL.

Le résultat de cette étape de traduction est illustré par les exemples présentés dans les tableaux 1.1 et 1.2.

	Original
Question	What is the UNGEGN, when the Value is 10 000 ?
Headers	['Value', 'Khmer', 'Word Form', 'UNGEGN', 'ALA-LC', 'Notes']
SQL	'SELECT UNGEGN FROM table WHERE Value = 10 000'

TABLE 1.1 : Exemple d'une question du jeu de données WikiSQL, ainsi que les entêtes de la table associée, et la requête SQL correspondante

	Traduction
Question	Qu'est-ce que l'UNGEGN, lorsque la valeur est de 10 000 ?
Entêtes	['Valeur', 'Khmer', 'Forme lexicale', 'UNGEGN', 'ALA-LC', 'Notes']
SQL	'SELECT UNGEGN FROM table WHERE Valeur = 10 000'

TABLE 1.2 : Exemple du résultat d'une traduction d'un item du jeu de données WikiSQL

Par ailleurs, nous avons respecté la partition du jeu de données original, à savoir, 56355 (70%) questions dans le jeu d'apprentissage, 8421 (10%) dans le jeu de validation et 1578 (20%) dans le jeu de test.

De plus, on peut appréhender la complexité des requêtes du jeu de données de deux manières : d'une part en observant la longueur des requêtes en langage naturel comme présenté dans le diagramme en figure 1.3, et d'autre part en observant les proportions des différents agrégateurs dans le jeu de données comme présenté dans le diagramme en figure 1.4.

On remarque alors que les requêtes en langage naturel sont assez courtes (autour d'une dizaine de mots). Cette distribution est semblable à celle du jeu de données anglophone présenté par [14]. De plus, on peut observer dans le diagramme en figure 1.4, que la majorité des requêtes SQL n'utilisent pas de fonctions d'agrégat, et ne s'assimilent donc qu'à une simple sélection sur la table. Ainsi ce jeu de données se caractérise par des requêtes d'une relative simplicité.

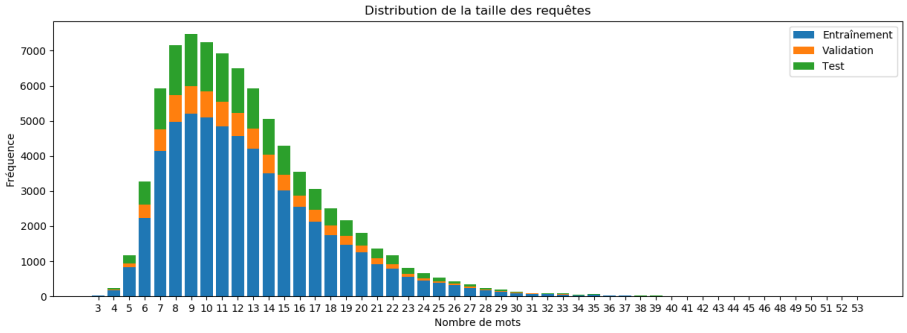


FIGURE 1.3 : Diagramme représentant la distribution des requêtes en français selon leur longueur

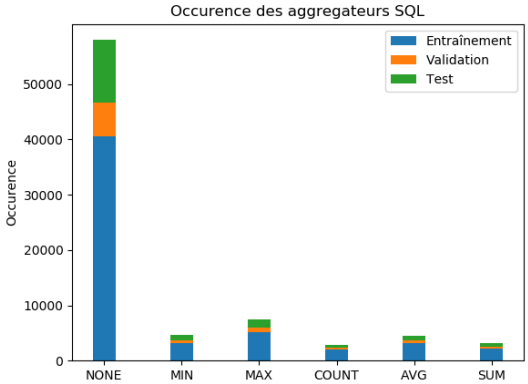


FIGURE 1.4 : Diagramme représentant la proportion de chaque agrégateur SQL dans les différentes partitions du jeu de données

Dans la section suivante, nous présentons l'architecture des transformers, un modèle neuronal profond qui constitue aujourd'hui l'état de l'art des modèles d'apprentissage automatique sur les tâches de Q&A.

1.4 Présentation de l'architecture des transformers

Jusqu'à récemment, afin d'analyser des phrases, l'état de l'art du traitement automatique du langage se concentrait sur des modèles conçus selon l'architecture *LSTM* (Long Short-Term Memory) [21]. Or, ces techniques présentent l'inconvénient d'analyser la phrase en considérant les mots de la phrase les uns à la suite des autres, suivant le sens de la lecture. Ceci entraîne plusieurs limitations dont certaines ont été levées via des *Bi-LSTM* [42], qui analysent la phrase dans les deux sens. Mais une vraie avancée fut apportée en 2017 par une nouvelle architecture proposée par Google [22], qui via un modèle d'auto-encodeur et un mécanisme d'attention, est capable de considérer l'influence de tous les mots simultanément sur l'ensemble de la phrase. Cette architecture a inspiré par la suite beaucoup de modèles, aux applications diverses [43, 44, 45] qui ont impacté l'état de l'art et tendent à devenir de plus en plus accessibles [46]. C'est le cas des modèles dont nous nous sommes inspirés et que nous avons utilisés dans nos travaux, nous prenons donc le temps ici de détailler le fonctionnement de cette architecture. [3].

1.4.1 Le mécanisme d'attention

Cette technique est inspirée de la manière dont nous focalisons notre vision sur certains mots pour lire une phrase. Par exemple dans la phrase : "Je conduis une voiture rouge" notre attention se porte plus volontiers sur la paire de mots ("conduis","voiture") ou la paire ("rouge", "voiture") que sur ("conduis","rouge").

Ces différences d'attention peuvent être dues à la fois aux liens sémantiques (entre "conduire" et "voiture"), et syntaxiques ("rouge" est ici l'adjectif de "voiture"). Le but du mécanisme d'attention est donc de modéliser l'influence mutuelle que peuvent avoir les mots d'une même phrase. A l'origine, cette technique fut employée par Bahdanau et al. [48] sur une architecture d'auto-encodeur pour aider à la tâche de traduction de l'anglais vers le français. La phrase n'y est plus représentée par un vecteur statique mais par un vecteur de contexte prenant en compte l'attention dans le plongement (l'embedding).

3. Un état de l'art plus détaillé de l'ensemble de ces méthodes a été réalisé par Oussama Ahmia dans le cadre de sa thèse [47]

FIGURE 1.5 : Schéma représentant comment l'attention se porte de manière différenciée selon les mots

$$c_i = \sum_{j=1}^{Tx} \alpha_{ij} h_i \tag{1.1}$$

$$\tag{1.2}$$

avec α_{ij} tel que :

$$\alpha_{ij} = Softmax(\alpha(S_{i-1}, h_j)) \tag{1.3}$$

$$\tag{1.4}$$

La fonction α représente ici la proximité entre le i-ème et le j-ème mot, en se fondant sur l'état caché S_{i-1}, et l'état h_j de l'encodeur. Cette fonction est déterminée par un apprentissage réalisé au même titre que les autres réseaux de neurones de l'architecture.

Cette technique nous intéresse principalement pour son utilisation dans l'architecture des *Transformers*.

1.4.2 Les Transformers

Présenté dans l'article *Attention is all you need*[22], *BERT* a profondément marqué l'état de l'art du traitement automatique du langage.

Les auteurs, Vaswani et al., utilisent une variante nommée *Scale Dot-Product Attention* du mécanisme d'attention décrit précédemment. Elle est définie par :

$$Attention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d_K}}V)$$

Avec :

$$Q = XW^Q \tag{1.5}$$
$$K = XW^K \tag{1.6}$$
$$V = XW^V \tag{1.7}$$
$$\tag{1.8}$$

Q, K et V désignant ici une requête (Q), un jeu de clés (K) et un jeu de valeurs (V).
Ce calcul est présenté en figure 1.6

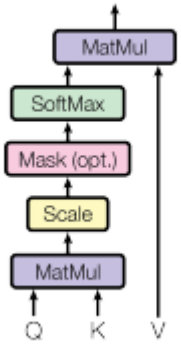


FIGURE 1.6 : Schéma représentant comment est calculée la *Scale Dot-Product Attention* [22]

Ici QK^T étant le produit scalaire des projections du vecteur de mot X dans les espaces W^Q et W^K, ce produit peut s'interpréter comme la corrélation entre un mot donné en entrée et les autres mots de la phrase. La projection de X dans W^V (V) permet ici d'adapter la taille du vecteur mot à une dimension d_v donnée.

La principale différence avec l'attention présentée précédemment est la division par un facteur $\frac{1}{\sqrt{d_k}}$, où d_k est la dimension du vecteur Q. Cette prise en compte de la longueur de la phrase a permis d'améliorer les performances du mécanisme d'attention (induisant un d_k grand).

Multi-Head Attention

Cette prise en compte de la longueur de la phrase a permis d'améliorer les performances du mécanisme d'attention

Enfin, les Transformers appliquent l'attention de plusieurs manières en simultané, en utilisant plusieurs "têtes". Chaque tête est caractérisée par un triplé de matrices (W^Q,W^K,W^V) entraînées indépendamment les unes des autres. L'embedding final est construit par concaténation des vecteurs d'attention des différentes "têtes". Cette architecture est présentée en figure 1.7. En cumulant différentes couches et en normalisant les embeddings de l'encodeur et du décodeur, on obtient l'architecture présentée en figure 1.8

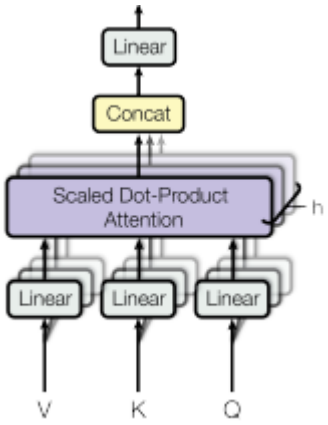


FIGURE 1.7 : Schéma de principe de l'attention multi-têtes (*Multi-Head Attention*) [22])

1.5 Adaptation d'un modèle anglophone à un modèle francophone, l'exemple des données tabulaires et du modèle TAPAS

1.5.1 Traduction des données

Notre choix s'est porté sur le système de traduction de Google (Google Translate API) afin de suivre le même protocole de traduction que celui proposé par [41] pour passer des données de Squad à SquadFr. Une étude récente réalisée par [49] a montré que l'outil est suffisamment performant pour notre cas d'usage, avec des scores selon les métriques BLEU [50] au delà de 88. Par ailleurs, nous ne nous intéressons ici qu'à la traduction de petites portions de textes relativement simples ; ce qui simplifie la tâche de traduction. Il serait néanmoins intéressant d'étudier plus spécifiquement l'impact de la traduction sur les performances du modèle et ceci pourra être envisagé dans des travaux futurs.

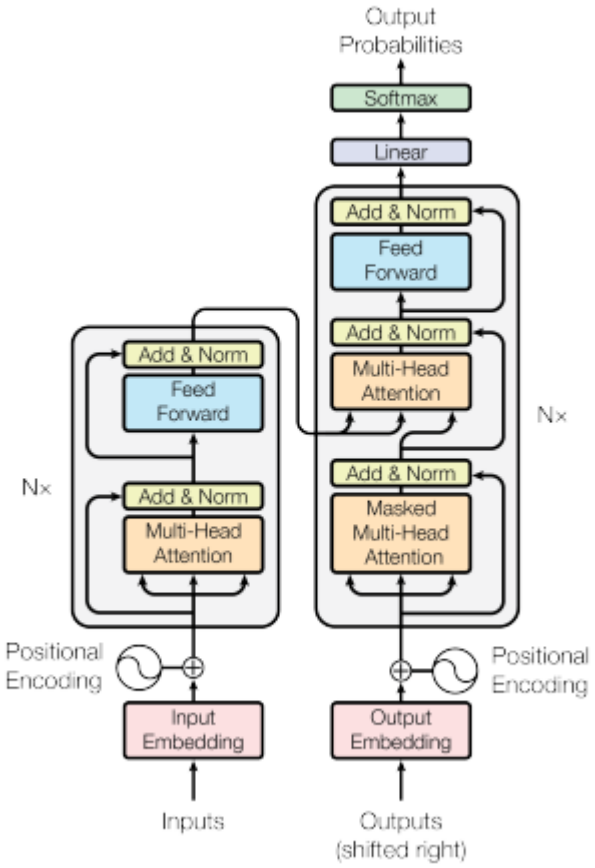


FIGURE 1.8 : Schéma représentant l'architecture de *Transformers* dans son ensemble [22])

1.5.2 Présentation du modèle

Si des progrès ont été faits sur le traitement du langage naturel, les services proposés restent cependant extrêmement limités car ils ne prennent en compte que des structures de données spécifiques, comme des portions de texte dans le cas de SQuAD, ou ils ne prennent en compte qu'un nombre pré-défini de requêtes. Au final, les interactions homme-machine restent encore très éloignées d'une interaction en langage naturel. Pour tirer partie de ces avancées prometteuses, une équipe issue de Google Research, composée de Jonathan Herzig et al. [36] a présenté TAPAS (Table parser), un modèle basé sur l'architecture BERT[38], en reprenant le même auto-encodeur, qui traite les questions et réponses comme des ensembles de données tabulaires. C'est cette inspiration qui permet au modèle TAPAS de scinder aisément son apprentissage en une phase de pre-training et une phase de fine-tuning, comme on peut le voir en figure 1.9.

Au lieu de créer un modèle contraint à une structure de table spécifique, les auteurs ont fait

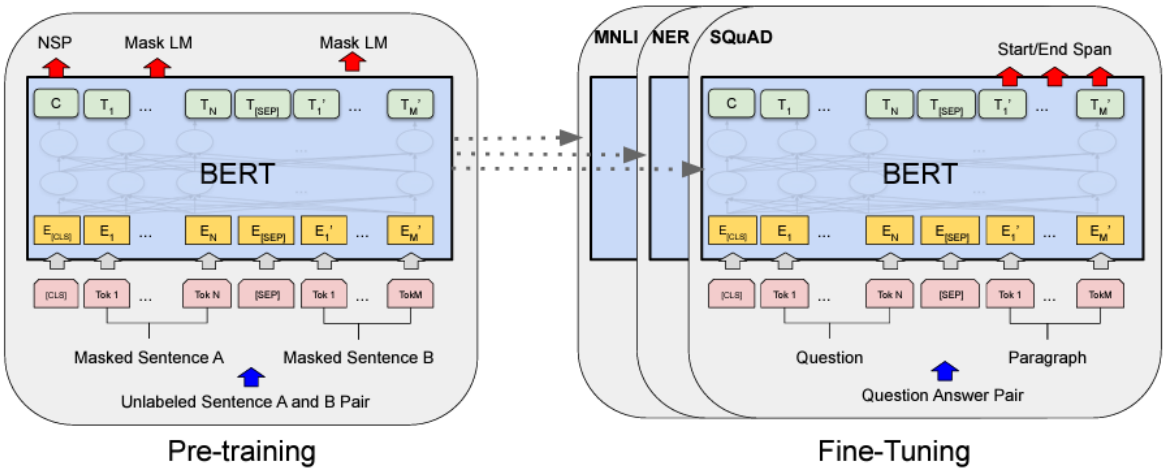


FIGURE 1.9 : Schéma représentant la procédure de pré-apprentissage (pre-training) et d'apprentissage fin (fine-tuning) de BERT [38]. Les mêmes architectures sont utilisées dans les deux cas, cependant le pre-training se focalise sur quelques paramètres, qui sont par la suite réutilisés pour le fine-tuning, ce dernier entraînant tous les paramètres suivant la tâche demandée.

le choix d'une approche plus globale en créant un réseau de neurones adapté à toute forme de jeu de données tabulaires. Le modèle TAPAS réutilise l'architecture de l'encodeur BERT, en y ajoutant des plongements supplémentaires pour encoder les informations structurelles des tables. En effet, si, bien souvent, les méthodes de QA passent par la traduction du texte en une requête SQL (comme décrit précédemment), TAPAS cherche ici directement la ou les cellules correspondante(s) ("cases" de la table), ainsi que l'aggrégateur correspondant (fonction s'appliquant au cellules sélectionnées, telle).

TAPAS exploite ces nouveaux plongements appris pour tenir compte des index de ligne et de colonne ainsi que d'un index de rang spécial qui représente l'ordre des éléments dans les colonnes numériques. L'architecture obtenue surpasse actuellement les autres modèles pour l'interrogation en langage naturel de données tabulaires.

TAPAS prend en entrée à la fois la question en langage naturel et la table associée sur laquelle on pose une question. Ce modèle est assez similaire à BERT mais il en diffère par l'ajout lors de la construction des plongements :

- Des positions relatives des mots dans la phrase ;
- De sept index sur les descripteurs des tables.

TAPAS est pré-appris sur une tâche de modèle masqué [4], à l'aide de 6.2 millions de tables venant de la version anglaise de Wikipedia et les textes correspondants.

4. Masked language modeling (MLM)

Enfin, TAPAS est entraîné plus finement afin de prédire deux informations, la première permet de sélectionner les cellules de la table, et la seconde de déterminer l'agrégation qui permet (éventuellement) d'effectuer des sommes ou des moyennes sur les cellules sélectionnées. Un schéma de synthèse de cette architecture est donnée en figure 1.10, ainsi qu'un exemple d'application en figure 1.11.

On peut voir alors que, dans cet exemple, le modèle pondère les valeurs des cellules selon leurs probabilités, et de même pour les aggrégateurs. Si ici la réponse attendue est un scalaire (somme de jours), cela peut aussi être la valeur d'une ou plusieurs cellules. Pour la sélection des cellules, le modèle commence par sélectionner un jeu de colonnes, puis les cellules.

Cet apprentissage concernant plusieurs aspects de la requête est permis par la fonction de perte, qui est une combinaison linéaire de trois fonctions, une pour la moyenne de l'entropie croisée binaire des colonnes, une qui calcule la même chose pour les cellules d'une colonne donnée, et une fonction dépendant de la probabilité de l'aggrégateur P_a.

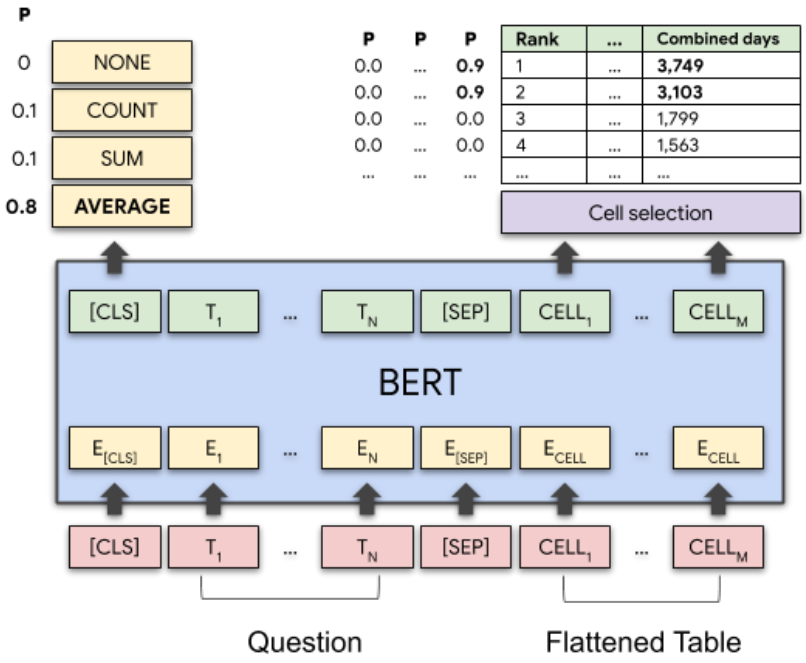


FIGURE 1.10 : Schéma représentant l'architecture globale du modèle TAPAS [36]

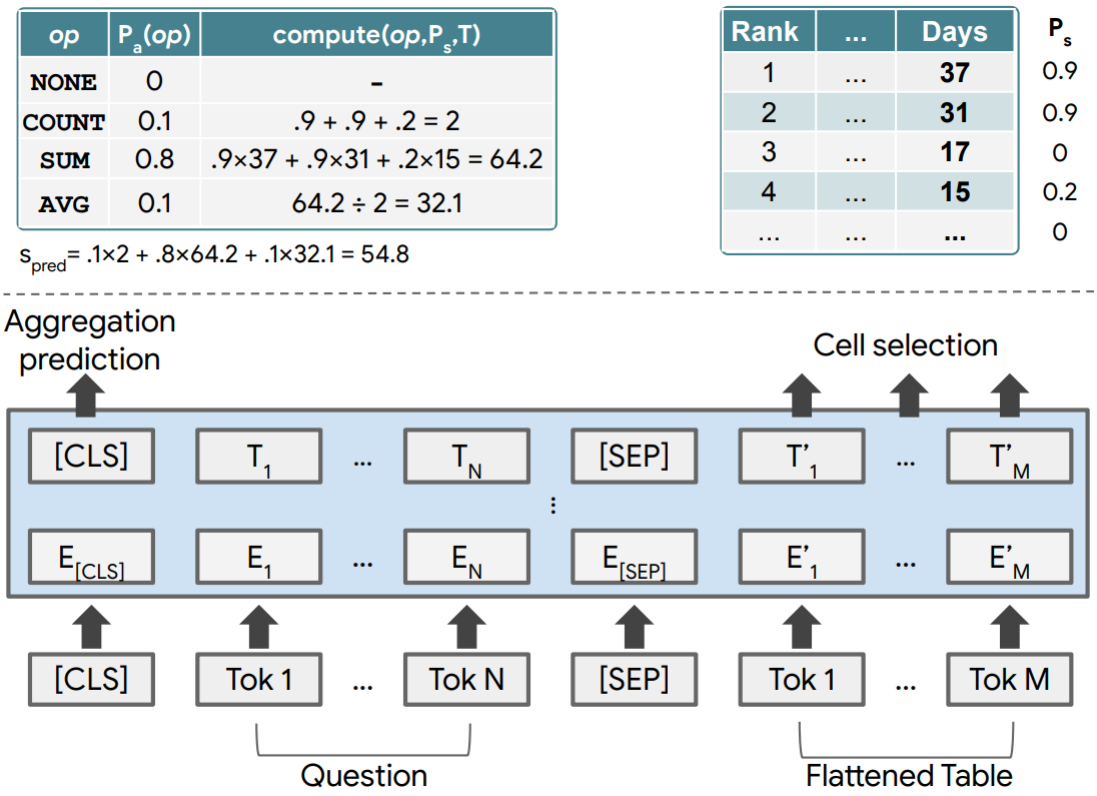


FIGURE 1.11 : Modèle TAPAS (en bas) appliqué sur la question "Total number of days for top two", prédiction de la cellule (à droite), donnée pour les colonnes selectionnées (en gras), et prédiction de l'aggrégateur (à gauche)[36].

1.6 Adaptation du modèle pour les données francophones, et expérimentation

1.6.1 Méthode pour adapter le modèle sur une tâche donnée (fine tuning)

Apprentissage du modèle

Outre les ajouts et modifications apportés au modèle BERT originel décrit précédemment, TAPAS suit un protocole d'apprentissage en trois étapes qui s'appuie sur plusieurs jeux de données. La première étape consiste en un pré-entraînement sur un jeu de données de 6,2 millions de tables anglophones extraites de Wikipedia suivant le modèle de masquage proposé dans BERT [38]. Le but ici est d'initialiser l'entraînement du modèle à partir du contexte constitué des éléments qui composent la table traitée : la tâche consiste à retrouver certains éléments masqués de ce contexte.

Cette étape de pré-apprentissage est suivie d'une étape d'ajustement fin (fine tuning) qui

finalise l'apprentissage du modèle sur une tâche spécifique. Les données utilisées pour cet entraînement sont alors similaires au jeu de tests pour évaluer le modèle final obtenu.

Dans leurs travaux les plus récents sur le modèle [37], les auteurs ont ajouté une étape intermédiaire de pré-apprentissage qui permet d'améliorer les performances [51]. Pour notre expérimentation, nous sommes partis du modèle ayant bénéficié du pré-entraînement supplémentaire.

1.6.2 Expérimentation sur les données francophones

Avant tout, il convient de préciser que l'approche utilisée ici est assez optimiste. En effet, dans une tâche de TAL, considérer qu'un modèle pré-entraîné sur une langue peut s'adapter à une autre langue durant le fine-tuning est une hypothèse forte. Cependant, dans notre cas l'aspect sémantique des données est autant dû au texte de la requête (relativement court) qu'à la structure des éléments dans la table qui, elle, est inchangée (hormis les mots en en-tête des colonnes et dans les cellules) ; c'est la raison pour laquelle nous nous autorisons de faire une telle hypothèse de travail.

De plus, nos travaux visent ici à déterminer si cette approche est satisfaisante, et dans quelle mesure une telle adaptation est coûteuse. Ainsi nous sommes parfaitement conscients que les résultats ont peu de chances d'être aussi performants que ceux du modèle original.

Le but de l'expérimentation est de déterminer dans quelle mesure ce nouveau jeu de données francophones impacte les performances du modèle TAPAS. Ainsi l'un des modèles TAPAS entraîné sur WikiSQL pour la langue anglaise sert de modèle témoin. À cela on compare un modèle suivant la même architecture et ayant reçu le même pré-entraînement, mais dont l'entraînement fin a été réalisé sur le nouveau jeu de données francophones.

Ces données sont sémantiquement les mêmes que celles du jeu d'origine, mais avec leurs tables et requêtes. Les tables sont traduites, afin de réaliser l'apprentissage, ainsi que les requêtes en langage naturel (traduites de l'anglais au français) afin de pouvoir tester notre modèle exactement dans les mêmes conditions que le modèle TAPAS d'origine, et ainsi permettre la comparaison de nos résultats.

Les deux modèles sont alors testés sur leurs jeux de données respectifs. Un schéma récapitule ce processus en figure 1.12, les métriques d'évaluation sont les mêmes que celles utilisées dans l'article originel de TAPAS. Les résultats obtenus sont dans le tableau 1.3.

Pour que les conditions expérimentales soient les plus similaires possible entre ces deux modèles, nous avons pris comme point de contrôle d'apprentissage le modèle *base* de TAPAS que nous avons entraîné sur les deux jeux de données respectifs avec 100000 et 200000 pas de batch 4 pour un total de 400000 et 800000 mises à jour. On obtient alors les résultats présentés dans le tableau 1.3. Les méthodes de test sont les mêmes que celles utilisées par TAPAS, et les modèles sont testés sur les mêmes données que celles utilisées pour l'apprentissage fin.

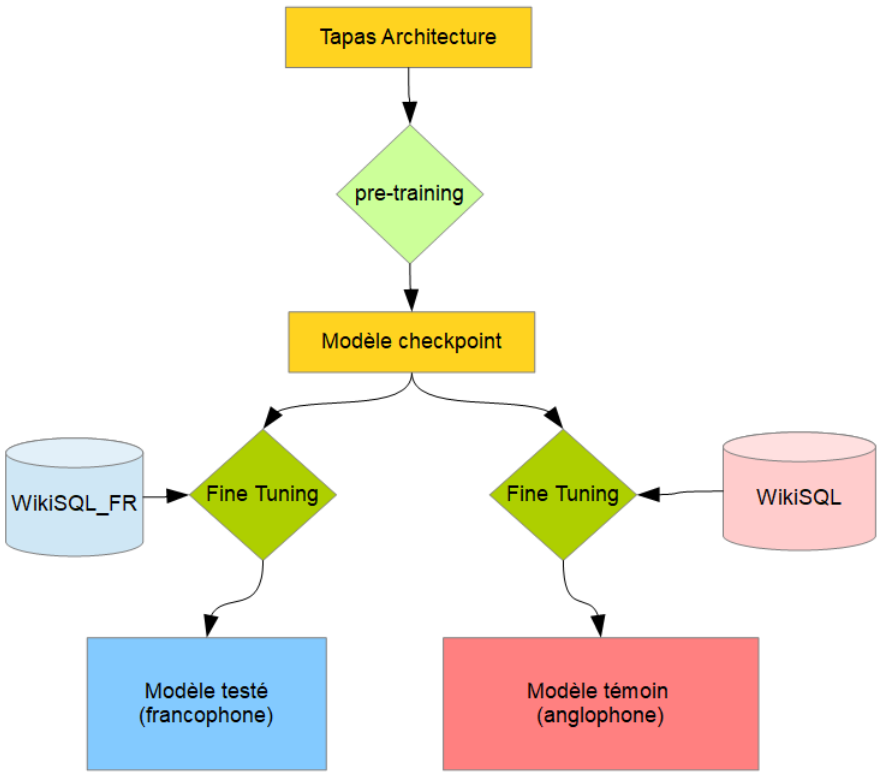


FIGURE 1.12 : Schéma représentant le processus d'obtention des modèles

	Modèle anglophone	Modèle francophone
100000 pas dev/test ex(acc)	0.8248/0.7993	0.5366 / 0.5195
200000 pas dev/test ex(acc)	0.8246 / 0.7979	0.6028 / 0.5864
résultats finaux[5]dev/test ex(acc)	0.8859	0.5967/0.5774

TABLE 1.3 : Exactitude (accuracy) sur le jeu de test et développement des deux modèles anglophone et francophone.

On peut constater plusieurs différences entre les résultats des deux modèles. Tout d'abord le modèle francophone semble obtenir, après 200000 itérations, des scores de performance 20% inférieurs à ceux du modèle anglophone. De plus le modèle de langue française semble bien plus sensible à cette dernière phase d'apprentissage que le modèle anglophone. Cette différence dans la vitesse de convergence de la deuxième phase peut s'expliquer par le changement de langue entre la phase de pré-apprentissage et la phase d'apprentissage fin, les résultats finaux présentés pour notre modèle francophone ayant atteint une valeur asymptotique. Un moyen de régler ce problème serait de faire le pré-apprentissage sur des données en français aussi, seulement, cette

5. Tels que présentés dans l'article de Eisenschlos et al. [37], après entraînement sur TPU.

méthode est très coûteuse en temps et en données, et hors de notre portée ici. De plus, nous ne pouvons pas utilisé la traduction automatique à chaque requêtes pour demander au modèle de travailler sur les requêtes en anglais. En effet, afin de conserver la main sur les données, nous ne pouvons nous permettre d'utiliser des APIs externes.

1.7 Conclusion et perspectives

L'interrogation de la base de donnée via une interface en langage naturel nous a semblé un bon point de départ. Cependant de part la difficulté d'appliquer les méthodes de Text2SQL nous nous sommes tournés vers un modèle "end-to-end", allant directement du langage naturel à la table, sans passer par une requête SQL. Ce type de modèle permet un ré-apprentissage sur nos données spécifiques : TAPAS

Ainsi nous avons voulu adapter le modèle proposé en réalisant un *fine-tuning* en utilisant un jeu de données traduit.

Les résultats de l'expérimentation ne permettent pas de conclure en une réelle utilité du modèle en situation d'exploitation. Néanmoins, nous avons pu ici tester une méthode d'adaptation d'un modèle anglophone vers un modèle francophone.

Cependant cette méthode est limitée comme le suggère la perte d'environ 20% des performances, par rapport aux performances du modèle entraîné sur l'anglais, ce qui n'est pas négligeable. Un apprentissage complet plutôt qu'un seul apprentissage fin permettrait probablement de pallier ce problème, mais le manque de données francophones adaptées à notre domaine, ainsi que d'architectures adaptées, posent de vraies limites.

Enfin, ce travail exploratoire nous a montré que le projet de créer une interaction entre l'utilisateur du CRM et les contacts via un chatbot relève d'une tâche qui n'est pas réalisable dans notre contexte. En effet, les approches par traduction en requête SQL nécessitent des données labélisées. Or, nous ne disposions pas dans le cadre de cette thèse de ressources pour créer de tels ensembles de données. De plus, cela nous éloigne du sujet principal de la thèse. En effet, ces travaux sur les requêtes en langue naturelle devaient être une étape préliminaire visant à assurer la qualité des réponses dans la relation client, avant d'analyser l'opinion des échanges par la suite.

En revanche, les utilisateurs *a priori* possèdent déjà un moyen de communication avec leurs contacts via les envois de mails. Ce constat sur la difficile faisabilité d'un chatbot sur des données tabulaires dans le cadre de cette thèse nous a conduit à orienter nos travaux sur l'étude des mails. Nous présentons nos investigations et les résultats obtenus dans les chapitres suivants.

BIBLIOGRAPHIE

[1] B. F. Green Jr, A. K. Wolf, C. Chomsky, and K. Laughery, "Baseball : an automatic question-answerer," in Papers presented at the May 9-11, 1961, western joint IRE-AIEE-ACM computer conference, 1961, pp. 219–224.

[2] A. Copestake and K. S. Jones, "Natural language interfaces to databases," The Knowledge Engineering Review, vol. 5, no. 4, pp. 225–249, 1990.

[3] J. Yang, A. Gupta, S. Upadhyay, L. He, R. Goel, and S. Paul, "Tableformer : Robust transformer modeling for table-text encoding," arXiv preprint arXiv :2203.00274, 2022.

[4] G. Obaido, "Phd thesis : Sql comprehension and synthesis," arXiv preprint arXiv :2203.03469, 2022.

[5] N. Deng, Y. Chen, and Y. Zhang, "Recent advances in text-to-sql : A survey of what we have and what we expect," arXiv preprint arXiv :2208.10099, 2022.

[6] I. Androutsopoulos, G. D. Ritchie, and P. Thanisch, "Natural language interfaces to databases–an introduction," Natural language engineering, vol. 1, no. 1, pp. 29–81, 1995.

[7] P. Cimiano and M. Minock, "Natural language interfaces : what is the problem ?–a data-driven quantitative analysis," in Natural Language Processing and Information Systems : 14th International Conference on Applications of Natural Language to Information Systems, NLDB 2009, Saarbrücken, Germany, June 24-26, 2009. Revised Papers 14. Springer, 2010, pp. 192–206.

[8] K. Affolter, K. Stockinger, and A. Bernstein, "A comparative survey of recent natural language interfaces for databases," The VLDB Journal, vol. 28, pp. 793–819, 2019.

[9] N. Nihalani, S. Silakari, and M. Motwani, "Natural language interface for database : a brief review," International Journal of Computer Science Issues (IJCSI), vol. 8, no. 2, p. 600, 2011.

[10] H. S. Dar, M. I. Lali, M. U. Din, K. M. Malik, and S. A. C. Bukhari, "Frameworks for querying databases using natural language : a literature review," arXiv preprint arXiv :1909.01822, 2019.

[11] F. Li and H. V. Jagadish, “Constructing an interactive natural language interface for relational databases,” Proc. VLDB Endow., vol. 8, no. 1, p. 73–84, sep 2014. [Online]. Available : https://doi.org/10.14778/2735461.2735468

[12] T. Mahmud, K. M. Azharul Hasan, M. Ahmed, and T. H. C. Chak, “A rule based approach for nlp based query processing,” in 2015 2nd International Conference on Electrical Information and Communication Technologies (EICT), 2015, pp. 78–82.

[13] T. Yu, R. Zhang, K. Yang, M. Yasunaga, D. Wang, Z. Li, J. Ma, I. Li, Q. Yao, S. Roman, Z. Zhang, and D. Radev, “Spider : A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task,” in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Brussels, Belgium : Association for Computational Linguistics, Oct.-Nov. 2018, pp. 3911–3921. [Online]. Available : https://aclanthology.org/D18-1425

[14] V. Zhong, C. Xiong, and R. Socher, “Seq2sql : Generating structured queries from natural language using reinforcement learning,” CoRR, vol. abs/1709.00103, 2017.

[15] Z. Yao, Y. Su, H. Sun, and W.-t. Yih, “Model-based interactive semantic parsing : A unified framework and a text-to-SQL case study,” in Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). Hong Kong, China : Association for Computational Linguistics, Nov. 2019, pp. 5447–5458. [Online]. Available : https://aclanthology.org/D19-1547

[16] Z. Yao, Y. Tang, W.-t. Yih, H. Sun, and Y. Su, “An imitation game for learning semantic parsers from user interaction,” in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). Online : Association for Computational Linguistics, Nov. 2020, pp. 6883–6902. [Online]. Available : https://aclanthology.org/2020.emnlp-main.559

[17] V. Zhong, M. Lewis, S. I. Wang, and L. Zettlemoyer, “Grounded adaptation for zero-shot executable semantic parsing,” in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). Online : Association for Computational Linguistics, Nov. 2020, pp. 6869–6882. [Online]. Available : https://aclanthology.org/2020.emnlp-main.558

[18] B. Wang, W. Yin, X. V. Lin, and C. Xiong, “Learning to synthesize data for semantic parsing,” in Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies. Online :

Association for Computational Linguistics, Jun. 2021, pp. 2760–2766. [Online]. Available : https://aclanthology.org/2021.naacl-main.220

[19] S. Iyer, I. Konstas, A. Cheung, J. Krishnamurthy, and L. Zettlemoyer, "Learning a neural semantic parser from user feedback," in Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers). Vancouver, Canada : Association for Computational Linguistics, Jul. 2017, pp. 963–973. [Online]. Available : https://aclanthology.org/P17-1089

[20] J. Ganitkevitch, B. Van Durme, and C. Callison-Burch, "PPDB : The paraphrase database," in Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies. Atlanta, Georgia : Association for Computational Linguistics, Jun. 2013, pp. 758–764. [Online]. Available : https://aclanthology.org/N13-1092

[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," Advances in neural information processing systems, vol. 30, 2017.

[23] B. Couderc and J. Ferrero, "fr2sql : Interrogation de bases de données en français," in 22ème Traitement Automatique des Langues Naturelles, 2015.

[24] P. Rajpurkar, R. Jia, and P. Liang, "Know what you don't know : Unanswerable questions for squad," 2018.

[25] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "Glue : A multi-task benchmark and analysis platform for natural language understanding," 2019.

[26] P. Liang, "Learning executable semantic parsers for natural language understanding," Communications of the ACM, vol. 59, no. 9, pp. 68–76, 2016.

[27] P. Pasupat and P. Liang, "Compositional semantic parsing on semi-structured tables," in Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1 : Long Papers). Beijing, China : Association for Computational Linguistics, Jul. 2015, pp. 1470–1480. [Online]. Available : https://www.aclweb.org/anthology/P15-1142

[28] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," Advances in neural information processing systems, vol. 27, 2014.

[29] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," arXiv preprint arXiv :1406.1078, 2014.

[30] M. Iyyer, W. tau Yih, and M.-W. Chang, "Search-based neural structured learning for sequential question answering," in ACL (1), 2017, pp. 1821–1831. [Online]. Available : https://doi.org/10.18653/v1/P17-1167

[31] D. Andor, C. Alberti, D. Weiss, A. Severyn, A. Presta, K. Ganchev, S. Petrov, and M. Collins, "Globally normalized transition-based neural networks," arXiv preprint arXiv :1603.06042, 2016.

[32] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields : Probabilistic models for segmenting and labeling sequence data," 2001.

[33] A. Neelakantan, Q. V. Le, and I. Sutskever, "Neural programmer : Inducing latent programs with gradient descent," arXiv preprint arXiv :1511.04834, 2015.

[34] P. Yin, Z. Lu, H. Li, and B. Kao, "Neural enquirer : Learning to query tables with natural language," arXiv preprint arXiv :1512.00965, 2015.

[35] T. Mueller, F. Piccinno, M. Nicosia, P. Shaw, and Y. Altun, "Answering conversational questions on structured data without logical forms," arXiv preprint arXiv :1908.11787, 2019.

[36] J. Herzig, P. K. Nowak, T. Müller, F. Piccinno, and J. Eisenschlos, "TaPas : Weakly supervised table parsing via pre-training," in Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Online : Association for Computational Linguistics, Jul. 2020, pp. 4320–4333. [Online]. Available : https://www.aclweb.org/anthology/2020.acl-main.398

[37] J. M. Eisenschlos, S. Krichene, and T. Müller, "Understanding tables with intermediate pre-training," 2020.

[38] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert : Pre-training of deep bidirectional transformers for language understanding," 2019.

[39] C. Baik, H. V. Jagadish, and Y. Li, "Bridging the semantic gap with sql query logs in natural language interfaces to databases," in 2019 IEEE 35th International Conference on Data Engineering (ICDE), 2019, pp. 374–385.

[40] Q. Lyu, K. Chakrabarti, S. Hathi, S. Kundu, J. Zhang, and Z. Chen, "Hybrid ranking network for text-to-sql," 2020.

[41] A. Kabbadj, "Something new in french text mining and information extraction (universal chatbot) : Largest qa french training dataset (110 000+)," November 2018, [Online ; posted 11-November-2018]. [Online]. Available : https://www.linkedin.com/pulse/something-new-french-text-mining-information-chatbot-largest-kabbadj/

[42] Z. Cui, R. Ke, Z. Pu, and Y. Wang, "Deep bidirectional and unidirectional lstm recurrent neural network for network-wide traffic speed prediction," arXiv preprint arXiv :1801.02143, 2018.

[43] L. Martin, B. Muller, P. J. Ortiz Suárez, Y. Dupont, L. Romary, de la Clergerie, D. Seddah, and B. Sagot, "Camembert : a tasty french language model," Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020. [Online]. Available : http://dx.doi.org/10.18653/v1/2020.acl-main.645

[44] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert : smaller, faster, cheaper and lighter," arXiv preprint arXiv :1910.01108, 2019.

[45] L. Floridi and M. Chiriatti, "Gpt-3 : Its nature, scope, limits, and consequences," Minds and Machines, vol. 30, no. 4, pp. 681–694, 2020.

[46] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz et al., "Transformers : State-of-the-art natural language processing," in Proceedings of the 2020 conference on empirical methods in natural language processing : system demonstrations, 2020, pp. 38–45.

[47] O. Ahmia, "Veille stratégique assistée sur des bases de données d'appels d'offres par traitement automatique de la langue naturelle et fouille de textes," Ph.D. dissertation, Lorient, 2020.

[48] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," arXiv preprint arXiv :1409.0473, 2014.

[49] M. Aiken, "An updated evaluation of google translate accuracy," Studies in Linguistics and Literature, vol. 3, p. p253, 07 2019.

[50] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu : a method for automatic evaluation of machine translation," in Proceedings of the 40th annual meeting of the Association for Computational Linguistics, 2002, pp. 311–318.

[51] J. Eisenschlos, S. Krichene, and T. Müller, "Understanding tables with intermediate pre-training," in Findings of the Association for Computational Linguistics : EMNLP 2020. Online : Association for Computational Linguistics, Nov. 2020, pp. 281–296. [Online]. Available : https://aclanthology.org/2020.findings-emnlp.27

Chapitre 2

La communication client par mail

2.1 Problématiques qualitatives et quantitatives des données.

Le chapitre précédent a montré que travailler sur des données tabulaires pour analyser la communication client ne semble pas être une bonne approche notamment du fait de la difficulté d'adapter des modèles au français tout en restant performant, et de la nécessité de construire de zéro de nouveaux canaux de communications via chatbots qui interrogeraient la base de données du CRM. Il nous faut donc choisir une autre source de données qui puisse être représentative de la communication client tout en étant exploitable. Les critères pour déterminer notre choix sont les suivants :

- **Quantité de données :** les données doivent être en nombre suffisant pour permettre a minima une analyse statistique. Un ordre de grandeur satisfaisant permettant d'envisager l'exploitation de méthodes "deep" learning en production pourrait se situer entre 500 et 1000 items.
- **Représentativité de la communication client :** on rappelle que l'entreprise *UNEEK* propose des services numériques à des structures telles que des entreprises, des écoles ou des associations. Or ces structures ont besoin de communiquer avec leurs contacts ou clients pour partager des informations ou demander un retour. Ainsi les données que nous exploiterons devront être issues d'un outil proposé par *UNEEK* qui permette cette communication client. De plus, l'analyse fournie doit pouvoir offrir un intérêt pour améliorer l'outil.
- **Possibilités d'analyser la qualité de communication :** pour pouvoir appliquer des méthodes d'analyse du langage naturel telles que par exemple l'analyse des émotions ou de l'opinion, nous avons préféré des documents aux données tabulaires.

2.2 La communication client

L'usage du traitement automatique du langage semble être un levier intéressant pour aborder les problématiques de relation et satisfaction client [1]. Ainsi l'analyse de la relation client se fait surtout par l'analyse de la satisfaction client, au travers de leurs réponses ou *feedback*.

Ce dernier peut se faire de 4 manières différentes :

- **Le traitement des réclamations et critiques d'un produit :** Bien souvent, les retours clients se font via de courts messages ou des commentaires liés à un produit. Ces messages sont souvent associés à une note globale, ce qui permet de quantifier ce retour. Cependant, il n'y a pas toujours de note associée et il convient alors de prédire la nature du commentaire ainsi que ce sur quoi il porte. Ainsi les outils du traitement automatique du langage pour l'analyse d'opinion ainsi que la reconnaissance d'entités nommées sont des tâches cruciales pour l'étude de ces données. De plus un jeu de données reconnu contient le retour de film avec les notes associées [1]. Ce jeu de données a été un excellent moyen pour l'état de l'art de comparer différentes méthodes. Cependant ce type de données a comme principal inconvénient de ne porter que sur des commentaires très courts et souvent sans contexte.

- **L'analyse d'emails :** Les retours clients peuvent également se faire par mail. Bien que plus rare, ce moyen de communication est toujours utilisé. Les mails ressemblent par de nombreux aspects aux commentaires, si ce n'est qu'ils sont souvent plus longs et destinés à un service en particulier. Ainsi leur analyse consiste en la prédiction de l'avis client et en la prédiction du thème du mail afin de rediriger ce mail vers le ou la personne en capacité de répondre au client concerné. De plus, les mails incitent davantage à une réponse de la part de l'entreprise, raison pour laquelle on cherche souvent à lier leur analyse à une génération automatique de réponses.

- **L'analyse des réseaux sociaux :** Les réseaux sociaux sont devenus très populaires et sont de fait un bon moyen de déterminer le profil des utilisateurs ou des clients. L'analyse de réseaux sociaux permet en effet d'avoir accès rapidement à une grande quantité de données de retours client associés à une marque, ou un produit. Cela permet ainsi d'avoir une idée globale de l'image de marque et de la satisfaction client [2]. Cependant, rassembler l'ensemble des opinions clients selon les réseaux sociaux demande de savoir gérer la détection d'ironie ou de sarcasme [3], ainsi que de prendre en compte les biais introduit par le focus sur ces communautés de client.

- **Les agents conversationnels :** Enfin, l'approche la plus souvent mise en œuvre pour automatiser la relation client est l'usage de chatbots, ou agents conversationnels [4]. Une des problématiques soulevées par ce système de communication, est de savoir à quel moment répondre. C'est pourquoi, beaucoup d'approches choisissent d'échanger en "tour par tour". Dans le cas contraire, cela nécessite un grand nombre de données pour pouvoir concevoir un modèle capable de faire une interaction homme-machine plus étendue. De

1. : https ://www.imdb.com

plus, avec l'augmentation de la complexité des interactions avec le client, il devient nécessaire d'avoir des modèles de TAL plus performants et plus de données qualitatives, pour avoir des chatbots fiables. En effet, les chatbots exposent à un risque non négligeable de mauvaise communication avec un client ou utilisateur, ce qui risque d'être dommageable pour l'image de l'entreprise [5, 6, 7]. Ce phénomène de mauvaise communication augmente d'ailleurs à mesure que les contacts du chatbot sont informés qu'ils communiquent avec une I.A [5, 7].

Ainsi l'utilisation de chatbots est certes privilégiée pour automatiser la communication client (d'où notre orientation de départ au chapitre 1). Mais ce moyen de communication demande à gérer des problématiques d'interaction homme-machine, tout en exposant le client à de possibles mauvaises expériences de communication, ce qui peut être contre-productif. C'est pourquoi dans notre cas, où nous cherchons à optimiser un service de communication client pour les entreprises clients de UNEEK, nous n'avons pas retenu cette approche.

Ainsi de tous ces moyens pour gérer la communication client, la grande majorité se focalise sur l'analyse et l'interprétation des messages émis par le client. Or, dans notre contexte, il semble plus pertinent de chercher à travailler sur la communication *émise* par l'entreprise à destination de ses clients. Enfin, parmi tous les canaux de communication, nous avons privilégié la communication par mail, car elle est encore beaucoup employée par les entreprises ou structures pour s'adresser à leurs communautés via des newsletters, et que cela peut s'insérer dans les moyens de communications actuellement gérés par UNEEK.

2.3 La communication par mail

Parmi les différents canaux possibles pour la relation client, et la communication d'informations, l'usage du mail est encore à la fois utilisé par 42,2 millions d'utilisateurs en France, pour 1,4 milliards de mails envoyés chaque jour[2], et celui représentant la plus grande ROI, avec une ROI 40 fois plus efficace que les réseaux sociaux selon l'Institut McKinsey [8]. De ce fait, s'intéresser à ce type de communication semble être un axe pertinent d'analyse de la relation client. Or les mails (et par extension les newsletters) comportent certaines spécificités [9].

Tout d'abord, l'email a des caractéristiques propres qui influent sur sa perception, et en premier lieu, l'objet du mail, qui est la première chose que voit ou lit le receveur du mail. D'autre part, le contenu du mail, qui constitue sa partie principale, influe aussi sur la perception de la newsletter par le client. Certaines études pointent qu'au sein de ce contenu, certaines caractéristiques telles que le placement des éléments [10] ou le design [11] influent sur les performances

2. Selon Médiamétrie pour janvier 2019 : `https://www.mediametrie.fr/sites/default/files/2019-02/2019%2003%2001%20CP%20Audience%20Internet%20Global_Janvier.pdf`

des mails. De plus, par rapport à une communication visuelle hors-ligne, telles que des affiches, Hoque et Lhose [10] estiment que la communication par mail dépend encore plus de la mise en page, du fait de la difficulté à lire sur un écran, et de la nécessité de devoir dérouler un message trop long.

Enfin, on peut aussi distinguer la nature du mail (commercial, informatif, etc) et de l'offre rattachée comme facteur influant sur sa perception.

Cependant, outre les caractéristiques propres à ce type de donnée qu'est le mail, il faut aussi considérer l'échange par mails comme une interaction sociale, et à ce titre, les caractéristiques de l'expéditeur (réputation, crédibilité, confiance, etc) ont un impact sur la perception du mail. On peut aussi noter l'influence des conditions de réception du mail (date et heure d'envoi, conditions de lecture, etc), mais aussi des critères sociaux du receveur (âge, sexe, éducation, etc).

Ces différentes influences sur l'attractivité d'un mail peuvent être mesurées à travers trois indicateurs de réponse de la part du receveur. Le premier, est l'ouverture du mail, qui permet de mesurer dans quelle mesure le champ "objet" du mail attire le lecteur, car c'est le seul élément à sa disposition à cette étape [3].

Dans un second temps, le taux de cliques (nombre de cliques divisé par le nombre d'ouvertures uniques) permet de mesurer l'attractivité du contenu du mail, que ce soit à travers ses caractéristiques propres (la forme du mail), ou la qualité de l'offre (le fond du mail).

Enfin l'indicateur final est la réponse définitive du lecteur. Dans le cas d'une newsletter, cela peut être un montant dépensé ou le nombre d'inscriptions. L'analyse de la qualité de cette réponse offre une mesure de l'attractivité perçue de l'offre seule.

Les analyses précédentes permettent de définir un modèle conceptuel d'une communication client résumée dans le schéma de la figure 2.1.

2.4 Établissement d'un jeu de données de newsletters

Choisir d'étudier la communication client par email décrite précédemment semble pertinent à plus d'un titre, mais parmi les multiples pistes possibles, il nous faut prendre en compte celles qui sont envisageables dans le contexte de l'entreprise *UNEEK*. *UNEEK* propose à ses clients un outil d'édition et d'envoi de newsletters à leurs contacts. Dans ce contexte, proposer un moyen d'optimiser la communication par mail consisterait à pouvoir prédire si la performance d'une newsletter est bonne ou mauvaise, avant son envoi. Un exemple d'une newsletter envoyée via l'outil de UNEEK est donné en figure 2.2.

S'il existe des méthodes d'optimisation d'envoi de mail, celles-ci se font surtout après l'envoi d'une partie des mails. La méthode la plus répandue est l'*A/B testing* [12]. Cette méthode

3. certaines messageries permettent d'avoir une pré-visualisation du contenu du mail, mais elles sont assez peu utilisées et pour la suite, nous considérons que seule la ligne d'objet est visible à l'ouverture.

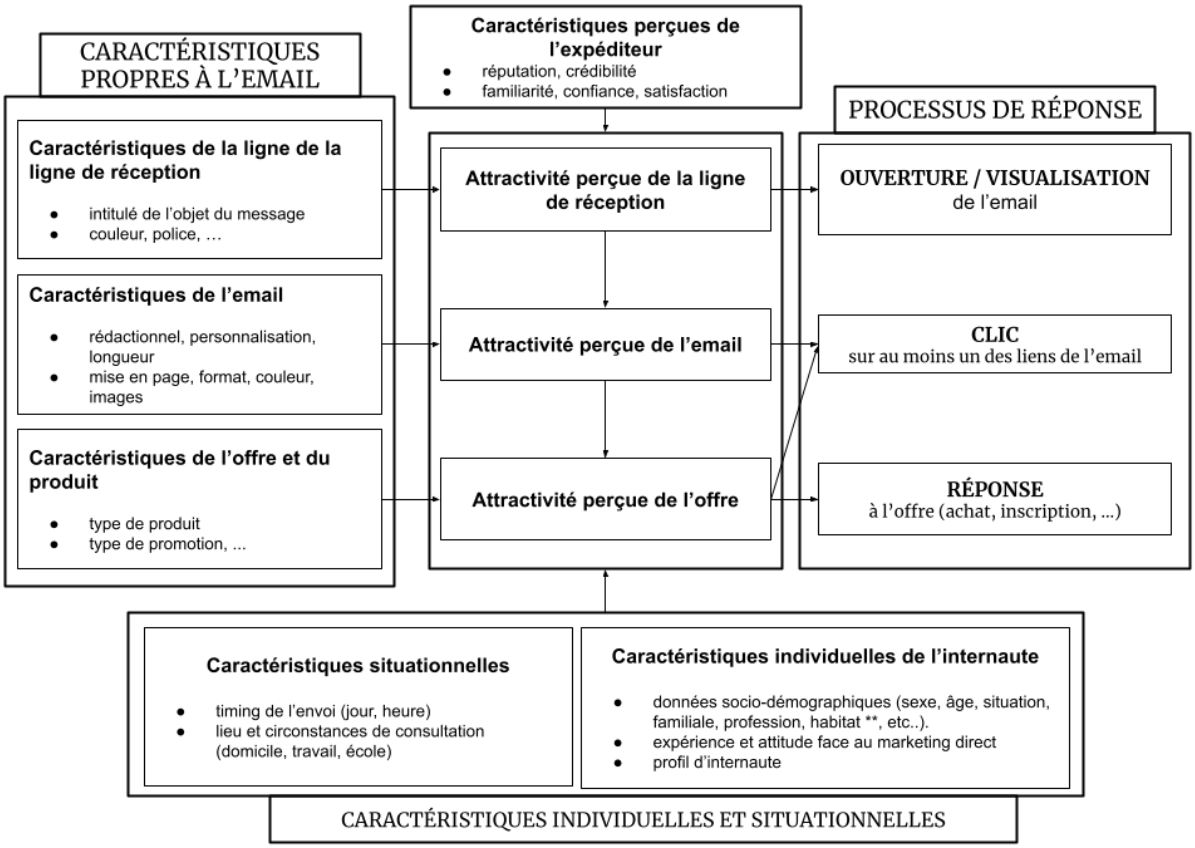


FIGURE 2.1 : Modèle conceptuel de la réponse de l'internaute face à la réception d'un email commercial [9]

repose sur l'envoi de deux modèles différents de la même newsletter, un de contrôle qui reprend souvent le modèle précédent et un second qui en diverge légèrement. Chaque modèle est envoyé à 50% des utilisateurs, les réponses des utilisateurs sont enregistrées et le meilleur modèle est conservé, comme illustré en figure 2.3. Le processus est alors répété aussi souvent que nécessaire. Cependant cette méthode est coûteuse sous plusieurs aspects. Tout d'abord, il faut 2 modèles de newsletters au minimum pour chaque communication, ce qui double le travail de l'éditeur de mail. Ensuite, le recueil des avis des utilisateurs ne peut se faire qu'après l'envoi, ce qui ralentit l'optimisation, et fait courir un risque sur la relation client auprès de 50% des abonnés, si jamais le modèle est mauvais.

C'est pourquoi la tâche ici consistera à prédire l'influence d'une newsletter avant son envoi, en se basant sur des caractéristiques décrites dans le modèle précédent. Il devient alors nécessaire de constituer un jeu de données pour pouvoir réaliser des techniques d'apprentissage, visant à

FIGURE 2.2 : Exemple de Newsletter envoyé par UNEEK, via son outil d'édition

la prédiction de l'impact de newsletters sur le lectorat. Pour cela nous utiliserons les données récoltées par *UNEEK* à la suite des envois de newsletters : à chaque campagne d'envois, le modèle de newsletter est enregistré.

2.4.1 Caractéristiques du mail

UNEEK sauvegarde les modèles de newsletters au format *.eml*, ce qui donne accès à plusieurs informations comme l'objet du mail et son contenu. Ainsi nous pouvons nous fonder sur ces deux

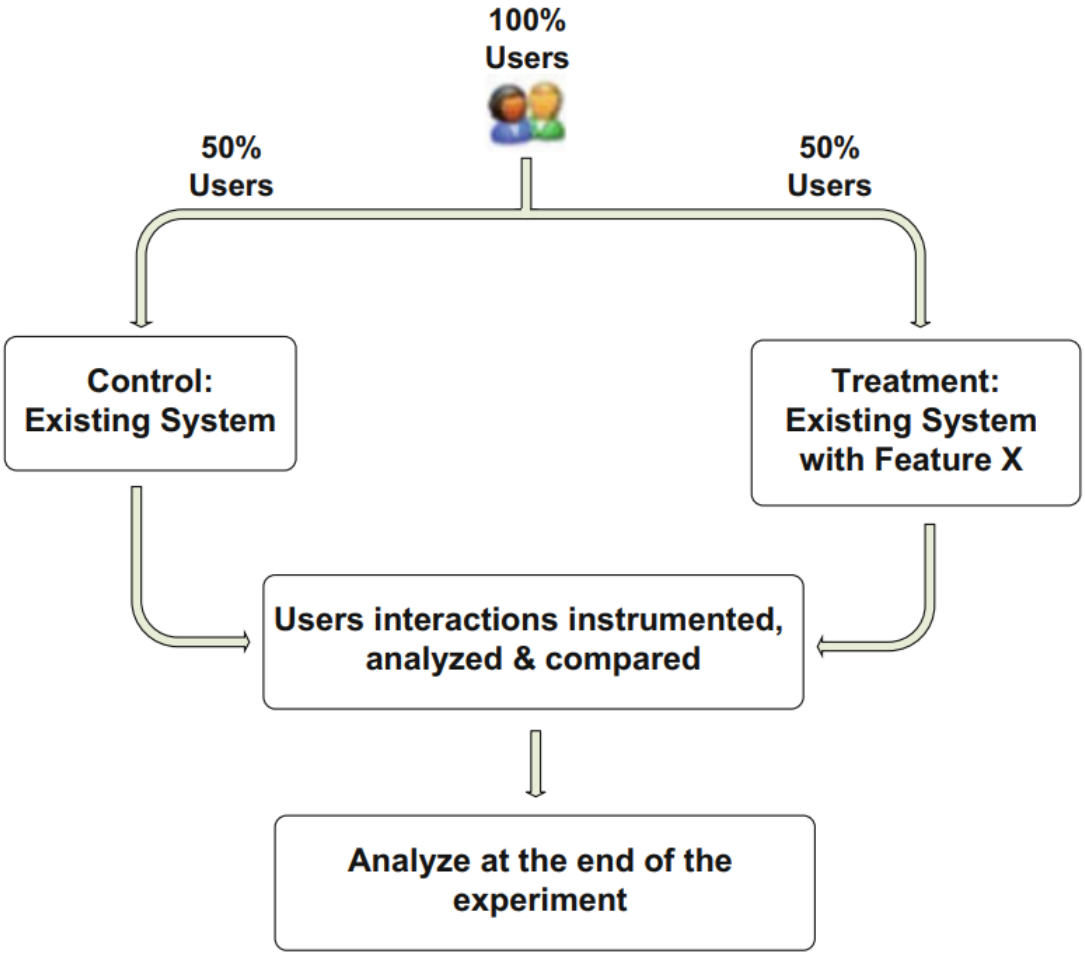


FIGURE 2.3 : Schéma décrivant le processus de l'*A/B testing* [12]

informations pour notre analyse prédictive. En revanche, il est difficile d'utiliser une description de l'offre proposée comme information pour notre jeu de données.

La difficulté est notamment due à l'hétérogénéité des sources des newsletters. En effet, un petit nombre de clients compose la majeure partie du jeu de données comme on peut voir dans le diagramme en figure 2.4. De plus les newsletters sont inégalement réparties parmi les clients. Ainsi analyser les thèmes abordés dans les newsletters risque d'introduire un biais trop important dans les données.

Les disparités évoquées forment cependant un biais négligeable si l'on excepte les thèmes abordés, les newsletters poursuivant toutes le même but global (fidéliser des inscrits et maximiser leur réactions). Par ailleurs, contrairement aux newsletters qui cherchent à vendre un produit, nos newsletters sont essentiellement informatives : il n'y a donc pas de critères liés au produit

ou aux conditions de vente à prendre en compte. Il serait en théorie possible de catégoriser ces newsletters suivant leur forme, mais la définition de "bulletin d'information" est assez floue et le travail de labélisation qui en découlerait serait coûteux pour un intérêt assez faible. Nous avons donc choisi de considérer notre jeu de données comme suffisamment homogène et ainsi, nous ne nous sommes concentrés que sur les caractéristiques directement extraites de l'analyse de la ligne d'objet et du contenu proprement dit du mail.

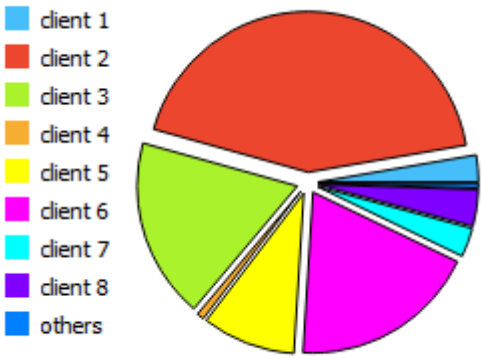


FIGURE 2.4 : Répartition des newsletters selon les clients de UNEEK les ayant envoyées.

2.4.2 Caractéristiques du receveur

Concernant les caractéristiques propres au receveur de mail, elles sont bien plus complexes à obtenir. En effet à la fois pour des raisons éthiques, légales (RGPD) et techniques il n'est pas possible dans le contexte de *UNEEK* d'effectuer un profilage complet des receveurs de mails. Si pour certains d'entre eux, il est possible de disposer des informations à travers un processus d'inscription, il n'est pas possible de déterminer quel receveur clique sur quelle campagne de newsletters, les informations sur les newsletters et les utilisateurs étant enregistrées de manières indépendantes. De plus comme précisé précédemment, les newsletters sont essentiellement à caractère informatif, ainsi la personnalisation du mail en fonction des conditions d'envoi (heure, date , etc) ne semble pas pertinente dans le cadre d'un bulletin mensuel ou d'une annonce d'évènement. Ainsi si ces caractéristiques du receveur comme décrit par Lancelot-Miltgen et al. [9], doivent jouer un rôle dans la perception des newsletters, ce ne sont pas des critères que nous conservons dans notre étude pour les raisons invoquées.

2.4.3 Caractéristiques de l'envoyeur

Les qualités perçues de l'envoyeur du mail et sa relation avec l'abonné jouent certainement un rôle dans l'attractivité d'une newsletter. Cependant traduire cette relation dans notre jeu de données est une tâche ardue. En effet, cela demanderait un travail général d'enquête auprès des abonnés pour connaître leur perception du client qui envoie le mail, et ce pour chacun des clients utilisant l'outil. Ce serait un sujet de travail intéressant, et demanderait un travail en sciences sociales complet, mais sort du cadre de cette thèse. Par ailleurs, un test de labélisation de l'attractivité du mail à travers une note de 1 à 7 a été fait sur un panel de newsletters sélectionnées de manière aléatoire au sein de l'entreprise *UNEEK*. Or il apparaît que l'accord inter-annotateur est trop faible pour rendre cette labélisation exploitable, alors même que les annotateurs avaient des profils similaires et travaillaient dans les mêmes conditions. Ces désaccords venaient notamment de la différence de priorité sur l'utilité perçue et le plaisir perçu [13], cette distinction entre ces deux perceptions de la newsletter est illustrée en figure 2.5.

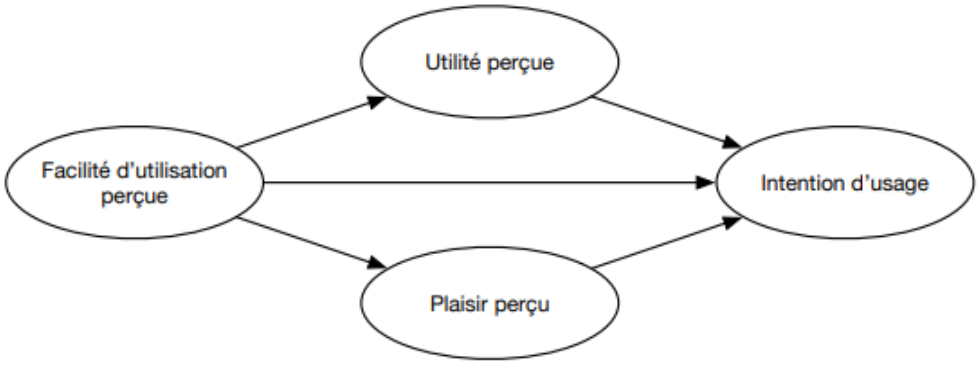


FIGURE 2.5 : Modèle d'intention d'usage intégrant le plaisir perçu [13]

Ainsi une newsletter souhaitant un joyeux noël sans autre but, peut être perçue très positivement par certains (marque d'attention, divertissement), et très négativement pour d'autres (nuisance, spam, inutilité). Or la tendance actuelle consiste de plus en plus à mêler les aspects informatif et de divertissement dans les communications par mail [14], cette confusion risque d'être de plus en plus grande et ce travail de labélisation encore plus complexe. En revanche on peut s'appuyer sur le contenu du mail pour analyser les émotions qu'il provoque et le ton employé. Nous décrivons ce point de manière plus approfondie dans le chapitre 3.

2.4.4 Réponses aux mails

Enfin le dernier type d'informations à prendre en compte est la réponse aux mails donnée par le receveur. Comme on l'a vu précédemment, on ne peut pas compter sur des indicateurs liés à

l'achat d'un produit, de par la nature des newsletters composant le jeu de données. De plus, on l'a vu, il est difficile de se fier à une annotation simple suite à une enquête de satisfaction suite à la réception de la newsletter. Il ne nous reste qu'à nous fier aux informations objectives recueillies par *UNEEK*, à savoir le taux de cliques, le taux d'ouverture et le taux de désabonnement. Ce dernier est cependant un événement trop rare pour être considéré dans notre jeu de données. Il faudrait un nombre bien plus important de newsletters pour compenser cette rareté.

2.4.5 Forme finale des données

Au final, notre jeu de données est d'environ 800 newsletters[4] suite à un n-ième filtrage éliminant les doublons et les newsletters de test sauvegardées dans la base de données. Ces dernières sont disponibles sous le format *.eml*, qui contient plusieurs informations dont la ligne d'objet et le contenu, ce dernier étant sous format *HTML*. Ce format est pour nous très intéressant, notamment pour pouvoir segmenter nos données. Enfin, toutes les newsletters envoyées par l'outil d'édition et d'envoi de newsletters de *UNEEK*, possèdent une ligne en haut du document, qui permet de compenser les éventuel bugs d'affichage liés à l'outil de messagerie. En cliquant sur ce lien, une fenêtre s'ouvre avec uniquement la newsletter affichée dans le navigateur. En utilisant un outil de capture d'écran automatique, il est alors possible d'avoir la représentation graphique de la newsletter sauvegardée. Cela nous sera utile, notamment dans le chapitre 4. Enfin, à chaque newsletter, on associe les indicateurs de performance i.e. les ouvertures et les cliques. Nous les normalisons ici afin de compenser les disparités entre les données. On prendra donc en compte le taux d'ouverture (nombre d'ouvertures uniques divisé par le nombre d'envois), et le taux de cliques (nombre de cliques divisé par le nombre d'ouvertures uniques).

2.5 Conclusion

Nous l'avons vu, les mails sont des moyens importants utilisés pour la relation et la communication client. À ce titre, chercher à optimiser l'envoi de mails permet de recentrer nos travaux sur une tâche précise, et offre de nombreuses perspectives. La première tâche a alors consisté à la construction d'un jeu de données exploitables, permettant une analyse prédictive de l'impact des newsletters. Pour ce faire nous nous sommes fondé sur les données sauvegardées par UNEEK via son outil d'édition d'envoi et de suivi de newsletters. Si bon nombre de critères influent sur la perception d'une newsletter, nous nous sommes restreints à ceux accessibles techniquement et pertinents quant à la nature des données. Toutes ces spécificités de nos données font que l'approche pour notre analyse se distingue des techniques d'email marketing plus couramment utilisées pour l'analyse prédictive [8]. Celles-ci se concentrent notamment sur la personnalisation du contenu et des conditions d'envoi en fonction d'un profilage du receveur, tandis que

4. Le nombre exact est 799 newsletters

nous cherchons ici à optimiser l'envoi en offrant à l'éditeur une prédiction de la réception *générale* de la campagne qu'il est en train de concevoir. Une fois ce jeu de données constitué, la question se pose de sa diffusion. Si a priori certaines newsletters ne posent pas de problèmes, beaucoup contiennent des informations personnelles ou sensibles : leur diffusion demanderait un gros travail d'anonymisation qui grèverait les données. Ainsi, si nous venons à diffuser le jeu de données, ce serait sous une autre forme, purement numérique. Par la suite nous verrons comment nous avons choisi d'exploiter au mieux ces données et leurs caractéristiques pour prédire les performances des newsletters en amont de leur envoi.

BIBLIOGRAPHIE

[1] Y. Piris and A.-C. Gay, "Customer satisfaction and natural language processing," Journal of Business Research, vol. 124, pp. 264–271, 2021.

[2] X. Liu, H. Shin, and A. C. Burns, "Examining the impact of luxury brand's social media marketing on customer engagement using big data analytics and natural language processing," Journal of Business research, vol. 125, pp. 815–826, 2021.

[3] S. Mukherjee and P. K. Bala, "Detecting sarcasm in customer tweets : an nlp based approach," Industrial Management & Data Systems, 2017.

[4] M. Mashaabi, A. Alotaibi, H. Qudaih, R. Alnashwan, and H. Al-Khalifa, "Natural language processing in customer service : A systematic review," arXiv preprint arXiv :2212.09523, 2022.

[5] C. Liu, J. Jiang, C. Xiong, Y. Yang, and J. Ye, "Towards building an intelligent chatbot for customer service : Learning to respond at the appropriate time," in Proceedings of the 26th ACM SIGKDD international conference on Knowledge Discovery & Data Mining, 2020, pp. 3377–3385.

[6] B. Sheehan, H. S. Jin, and U. Gottlieb, "Customer service chatbots : Anthropomorphism and adoption," Journal of Business Research, vol. 115, pp. 14–24, 2020.

[7] Á. Aldunate, S. Maldonado, C. Vairetti, and G. Armelini, "Understanding customer satisfaction via deep learning and natural language processing," Expert Systems with Applications, vol. 209, p. 118309, 2022.

[8] R. ABAKOUY, A. El HADDADI, E.-N. El Mokhtar, F. LIST, and E. DMI, "L'analyse prédictive des campagnes email marketing avec le big data mining."

[9] C. Lancelot Miltgen, Y. Costes, T. Munier, and S. Gauthier, "L'efficacité d'un e-mail à vocation commerciale : étude de l'influence des caractéristiques sociodémographiques des internautes sur le processus de réponse," Revue Française du Marketing, vol. 205, pp. 21–40, 2005.

[10] A. Y. Hoque and G. L. Lohse, "An information search cost perspective for designing interfaces for electronic commerce," Journal of marketing research, vol. 36, no. 3, pp. 387–394, 1999.

[11] A. Ansari and C. Mela, "E-customization," Journal of Marketing Research - J MARKET RES-CHICAGO, vol. 40, pp. 131–145, 05 2003.

[12] R. Kohavi and R. Longbotham, "Online controlled experiments and a/b testing." Encyclopedia of machine learning and data mining, vol. 7, no. 8, pp. 922–929, 2017.

[13] G. Gronier, "Les émotions face à l'usage des technologies de l'information et de la communication en entreprise," C. Berghmans. Intelligence et compétences émotionnelles en entreprise : perspectives multiples, pp. 290–310, 2018.

[14] M. S.-F. Virginie Rodriguez, "Le contenu des communications relationnelles par email des enseignes : Quelle perception par le consommateur ? [Content of retailers' relational e-mails : what is the consumer's perception ?]," in 20th International Marketing Trends Conference, Venise, Italy, Jan. 2021, Accessed on Aug. 29, 2021.

Chapitre 3

Analyse d'Opinion et d'Émotion

3.1 L'analyse d'opinion et d'émotion, une problématique large

Le besoin et les opportunités d'améliorer la communication en vue d'améliorer la relation ont été décrits dans le chapitre 2. Or, de plus en plus de gens publient des tweets, blogs, vidéos, etc. pour exprimer leurs sentiments et opinions. Leur analyse automatique constitue un enjeu sociétal et commercial important qui explique l'intérêt grandissant pour l'analyse d'opinion ou d'émotions en traitement automatique du langage (TAL). Dans notre cas, l'objectif visé est l'amélioration de la relation client dans les campagnes d'emails.

Cependant le sujet est vaste et difficile. En TAL, sa sensibilité au contexte et au domaine et l'importance de la forme expressive (négations, humour, choix des mots, etc.) sont de véritables défis. Par ailleurs, au delà du TAL, ce sujet concerne également les interactions homme-machine, les neurosciences et les diverses théories psychologiques sur les émotions présentes dans le texte.

Dans cette thèse nous essayons de présenter un état de l'art de la modélisation des émotions pour notre tâche d'optimisation de la relation client. Nous pouvons dès à présent dire que le fait même de définir ce qu'est une émotion de manière formelle n'est pas aisée et que tous les modèles que nous présenterons se fondent sur des hypothèses de travail qui peuvent être discutées d'un point de vue psycholinguistique.

Prendre en compte toutes les critiques de ces modèles demande un très gros travail pluridisciplinaire de modélisation et de données de l'état de l'art pour tenter de trouver un consensus, ce qui n'est pas l'objet de cette thèse. Ainsi, nous nous contenterons de présenter les modèles les plus couramment utilisés dans la littérature du traitement automatique du langage, ainsi que leurs hypothèses et limites. De plus les termes de la littérature anglo-saxonne peuvent prêter à confusion. Ainsi, les termes « opinion mining » et « sentiment mining » sont souvent interchangeables. À l'instar des expressions « emotion detection », « affective computing » et « emotion analysis » qui désignent la même idée [1].

Ici, nous désignerons par « détection d'opinion » et « analyse d'opinion », les tâches de détection et d'analyse de l'orientation d'un texte suivant deux critères d'analyse binaires ; à savoir selon un axe positif/négatif et objectif/subjectif. Par ailleurs, nous utiliserons les termes « détection d'émotions » et « analyse d'émotions » pour désigner les tâches de détection et d'analyse des émotions au sein d'un texte reposant sur des modèles qui prennent en compte un

jeu d'émotions plus varié.

3.1.1 L'analyse d'opinion

Comme décrit plus haut, cette tâche consiste à déterminer si le texte est globalement positif ou négatif ; très utilisée pour l'analyse de tweets, cette tâche est bien documentée et dans notre cas nous avons utilisé les méthodes incluses dans la bibliothèque TextBlob de Python.

Ces méthodes permettent d'attribuer deux valeurs à un texte donné. La première est une mesure de la *polarité*. Prenant valeur dans l'intervalle $[-1; 1]$ elle détermine ici si ce texte est plutôt joyeux et positif (1), ou au contraire triste et négatif (-1).

La seconde est une mesure de la *subjectivité* qui prend valeur dans l'intervalle $[0; 1]$, la phrase étant considérée comme portant un avis objectif à 0, et à l'inverse, un propos personnel et subjectif à 1.

Cette méthode de traitement automatique du langage a fait l'objet d'une implémentation dans le langage Python [2, 3], utilise un modèle intégré d'analyse du français pour déterminer la subjectivité et la polarité du texte. Cette méthode de la bibliothèque Textblob est initialement conçue pour de l'anglais, mais une autre bibliothèque du même auteur : textblob-fr [1], adapte ces techniques au français.

Cela est en effet possible, car les techniques de POS-tagging et d'analyse d'opinion se fondent sur les travaux de Tom de Sedt et Walter Daelemans [4] qui proposent ces techniques dans différentes langues, notamment en adaptant le lexique qui constitue la base de cette approche. Pour le français, cette adaptation se fonde notamment sur les corpus LEFFF [5] et LEXIQUE [6].

Plus précisément pour l'analyse d'opinion proprement dite, la technique consiste à considérer des adjectifs avec une valeur qui leur est attribuée en positivité, subjectivité, intensité et confiance. Grâce à ces valeurs et en prenant en compte l'effet des adverbes et des négations, il est possible d'en déduire un tuple (polarité, subjectivité) pour toute une phrase. Via cette méthode les auteurs obtiennent un taux de bonne classification (accuracy)[2] de 75% sur des commentaires de livres en français [3]. Des détails sur l'utilisation des adjectifs pour l'analyse d'opinion sont donnés par les mêmes auteurs à propos d'un lexique en Néerlandais [7]. Les auteurs y présentent notamment comment ces deux valeurs, la positivité et la subjectivité, sont complémentaires dans l'analyse d'opinions. Ce modèle peut être visualisé à travers la figure 3.1.

Ces deux valeurs combinées permettent d'obtenir une représentation de la phrase en lien avec l'opinion et le discours de son auteur. Cependant, si comme dit plus haut, cette approche permet d'analyser assez bien des tweets et des retours clients, elle semble désormais limitée. Plusieurs travaux cherchent en effet à offrir une analyse plus détaillée et complémentaire via la

1. https ://github.com/sloria/textblob-fr
2. par la suite nous utiliserons le terme anglais d'accuracy pour désigner cette métrique
3. https ://github.com/clips/pattern/blob/master/pattern/text/fr/fr-sentiment.xml

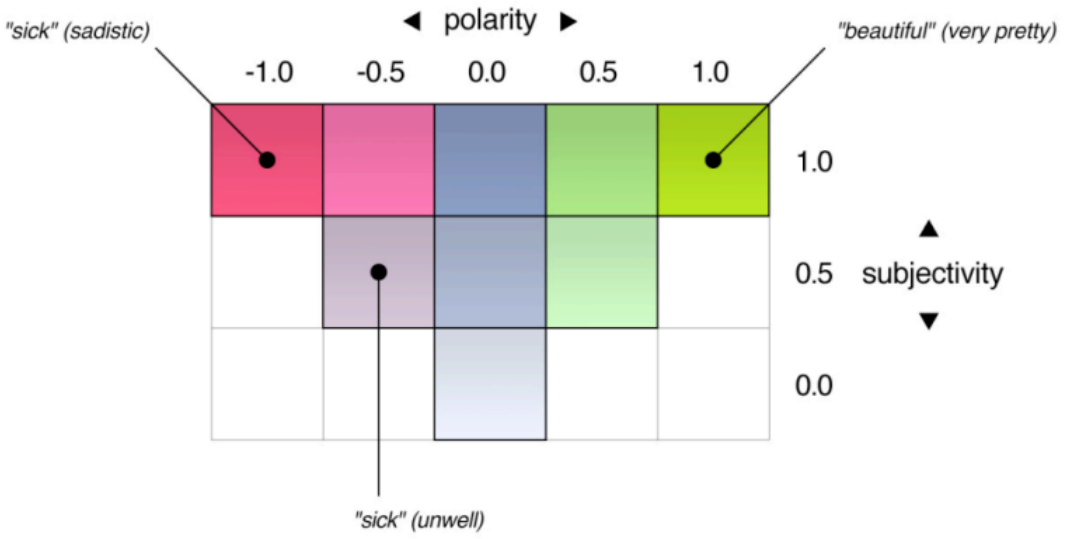


FIGURE 3.1 : Représentation graphique de l'opinion des mots suivant les axes de polarité et subjectivité [7]

détection à l'analyse d'émotions. Un exemple d'une telle application peut être revue à travers l'outil Watson développé par IBM qui se propose d'analyser le ton d'un texte [8]. [4].

Une des motivations d'une telle approche, semble être la volonté de garantir une interprétation des émotions transmises par le texte plus proche de l'interprétation humaine.

De plus, cette analyse de l'opinion fait aussi partie de la problématique plus large de l'analyse d'aspect (*Aspect Based Sentiment Analysis ou ABSA*). Cette tâche consiste notamment en l'analyse des retours des avis clients sous différents aspects. On peut diviser l'ABSA à l'échelle d'une phrase en trois sous-tâches : [9]

- Extraction de la cible de l'opinion(*Opinion Target Extraction ou OTE*). Cette tâche consiste en l'extraction de l'expression linguistique utilisée dans le texte se référant à une paire d'entité E et d'attribut A pour lesquels une expression est exprimée dans une phrase donnée. Par exemple dans la phrase "un bon repas", l'entité "repas" est associée à l'attribut "bon".

- Détection de la catégorie d'aspect(*Aspect Category Detection (ACD)*. Cette tâche consiste en l'identification d'une entité E et d'un attribut A définis selon un jeu de catégories définies en amont selon le contexte. Comme par exemple "nourriture" ou "service" pour des avis sur un restaurant.

4. Le 24 février 2022, IBM a transféré l'API de Watson™ Tone Analyzer vers une API comprenant plus largement plusieurs applications de NLU (Natural Language Understanding)

- Polarité de l'opinion (*Sentiment opinion polarity*). Pour chaque paire entité-attribut (E A), on cherche alors à déterminer si l'opinion exprimée est positive, négative ou neutre.

Tout ces processus sont visibles en figure 3.2, à l'aide un exemple en anglais.

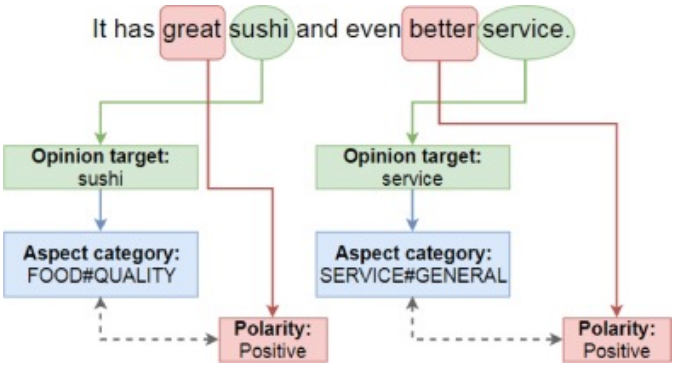


FIGURE 3.2 : Trois tâches de l'ABSA à travers une phrase d'exemple venant du jeu de données du SemEval ABSA dataset 2016. La phrase à deux cible d'opinions, "sushi" et service. La catégorie "sushi" de sushi est "Food" avec pour attribut "Quality" et pour polarité "Positive". Pour le "service" on y associe la catégorie "Service" et une opinion elle aussi "Positive"
[9, 10]

Il existe des tâches d'ABSA à l'échelle du texte, et au sein de domaines ouverts. Cependant les tâches sur les phrases sont plus précises et détaillées. De plus certaines approches utilisant de l'apprentissage profond sur ces tâches ont été testées [10]. Cependant plusieurs études robustes semblent être performantes seulement pour l'extraction et la catégorisation de l'aspect ; à l'inverse celles s'intéressant à la détection de l'aspect et l'analyse de sentiment n'ont pas atteint des scores de performance optimaux.

L'ABSA est ainsi très utile notamment pour contextualiser et interpréter un retour client. Cependant, dans notre cas, nous nous intéressons à une communication *vers* le client par le biais de newsletters. De plus si cela permet d'apporter un contexte à l'opinion, et donc de l'ancrer dans un domaine lié au document, cela reste limité à une approche de polarité positive/neutre/négative.

3.2 Les différents modèles d'émotions dans le texte

3.2.1 Modèles catégoriels

On appelle ici modèle catégoriel (traduction de *categorical models*) les modèles qui analysent les émotions à travers un jeu d'émotions prédéfini. L'hypothèse de ces approches est de considérer que toutes les émotions humaines peuvent être décomposées en certaines émotions dites « basiques ». Cette approche peut être qualifiée de discrète car elle consiste, dans le cadre du traitement automatique du langage, à attribuer à un mot ou une phrase, un label d'une ou plusieurs

émotions basiques. Dès lors la problématique consiste, en premier lieu, à définir quelles sont les émotions que l'on peut qualifier de basiques. Des études comparatives permettent d'avoir une vue d'ensemble assez complète de ces modèles [11, 1, 12, 13, 14, 15, 16, 17], et l'on peut voir qu'il en existe un grand nombre avec un nombre de catégories pouvant varier énormément. Faire une liste exhaustive des modèles ou des émotions basiques retenues n'est pas le but ici. Cependant, certaines approches ont plus marqué la littérature de l'analyse des émotions que d'autres et ce sont celles-ci que l'on va détailler.

Le modèle d'Ekman

Pour déterminer quelles sont les émotions de base, Paul Ekman s'est appuyé sur les expressions du visage. En effet ses travaux semblent montrer que, dans plusieurs cultures tout autour du monde, les expressions faciales permettent de reconnaître des émotions caractéristiques. Ainsi, la joie, la peur, la colère, la tristesse, le dégoût et la surprise semblent être universellement reconnues sur les visages humains, comme on peut le voir dans la figure 3.3.

On peut alors remarquer que de ces émotions, il y a quatre émotions négatives, une neutre et une positive. Ce déséquilibre peut s'expliquer selon Ekman par la plus grande nécessité, d'un point de vue anthropologique, de pouvoir détecter les émotions négatives sur le visage d'autrui que ce soit en prévision d'un danger, ou pour un besoin social d'empathie. Cependant, un tel déséquilibre entre les émotions positives et négatives constitue pour certains une faiblesse de ce modèle car il offrirait une sur-représentation des émotions négatives tandis que les aspects positifs sont uniquement représentés par la « joie » sans plus de détails. De plus même si ce modèle s'appuie sur une étude anthropologique sérieuse, dire que ces 6 labels sont universels pour les émotions ne serait pas exact. D'ailleurs Ekman lui-même [19], précise que ces 6 émotions sont censées englober plus que ce que nous entendons par les termes associés, leurs sémantiques n'étant jamais exactement la même suivant les langues et les cultures.

Toutefois, bien souvent, le modèle d'Ekman forme une base de travail pour des modèles plus complexes, qui utilisent des catégories d'émotions différentes. Ainsi dans bon nombre de modèles, on retrouvera des émotions présentes dans les 6 émotions proposées par Ekman.

Le modèle Narasava

Le modèle catégoriel d'émotions appelée *Narasava* est originaire de la tradition indienne du *Nâtya-shâstra*[5] et il est composé de 9 émotions. Contrairement au modèle d'Ekman, ce modèle propose autant d'émotions "positives" que "négatives". Utilisé dans le cadre de détection d'émotions dans le théâtre et la poésie indienne [14], ce modèle reprend des émotions introduites par Ekman (la colère la peur la joie la tristesse et la surprise), en supprime certaines (le dégoût), et

5. Une version codifiée du théâtre indien, que nous ne détaillerons pas ici, bien que intéressante.

FIGURE 3.3 : Visages représentant les faciès présentant les 6 émotions d'Ekman utilisées par Paul Ekman pour définir les émotions de bases [18]. De gauche à droite et de haut en bas : la joie, la surprise, la peur, la colère, le dégoût et la tristesse

en rajoute de nouvelles (la paix, la haine, le courage, l'amour). Cependant, ce modèle a été conçu dans un but très spécifique [6] et, même si cette approche peut présenter des avantages, elle est difficilement applicable dans d'autres champs d'application. Ce modèle peut donc être vu comme une extension ou une modification du modèle d'Ekman pour s'adapter à un contexte particulier d'études du langage. C'est ainsi que beaucoup d'autres modèles catégoriels particuliers divergent par rapport au modèle à six émotions d'Ekman. Ces modifications ont pour but de s'adapter à un contexte particulier, en ajoutant ou supprimant des émotions de base en fonction de ce qui semble plus approprié compte tenu du contexte applicatif.

6. la détection d'émotions dans la poésie indienne

3.2.2 Modèles dimensionnels

Une approche différente des modèles catégoriels consiste à représenter les émotions au sein d'un espace dont les dimensions sont définies par des mesures de caractéristiques jugées fondamentales des émotions. Cela permet, contrairement aux modèles catégoriels, d'identifier les dépendances mutuelles entre les émotions et leurs points de convergence et de divergence. Ainsi, selon ces modèles, toute émotion peut être vue comme un point présent dans l'espace ainsi formé par les axes des émotions de "base". L'application de ce modèle consiste donc à associer à un texte ses coordonnées selon les axes émotionnels du modèle. Ainsi nous verrons ici comment de tels espaces de représentation des émotions sont définis et créés.

Le modèle circomplexe

La plupart des modèles dimensionnels prennent en compte certaines mesures pour former les axes des dimensions. Ainsi, une des approches les plus souvent reprise est celle présentée par Russel [20, 21, 22], est nommée modèle circomplexe (*model circumplex*). Ce dernier propose deux composantes principales des émotions : la valence (ou positivité), et l'éveil ou énergie (*arousal*). Ces deux axes permettent alors de recomposer des émotions, que l'on place dans le plan ainsi formé, comme illustré en figure 3.4.

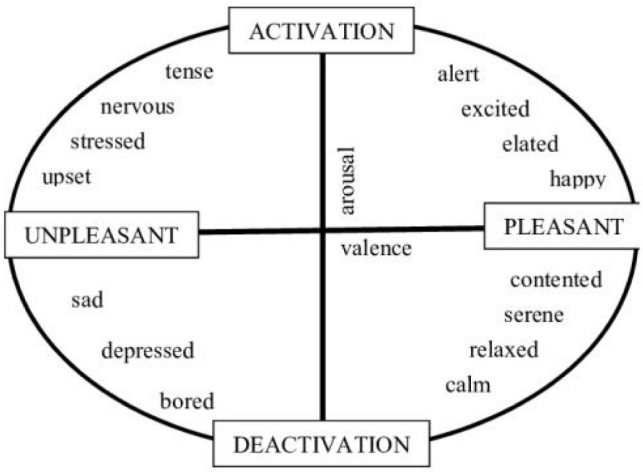


FIGURE 3.4 : Modèle circomplexe de Russel [20, 16]

Toutefois, Russel et Mehrabian [23], ont proposé une évolution du modèle en y ajoutant une troisième dimension, les émotions se répartissant alors selon les trois axes : plaisir-déplaisir (*pleasure*), excitation-non-excitation (*arousal*), et domination-soumission (*dominance*). Les auteurs appuient leur propos, en faveur de ces trois axes, selon le résultat de plusieurs études. Par la suite cette approche en trois dimensions (PAD) a été amplement étudiée.

Le modèle PAD *Pleasure-Arousal-Valence*

Si les approches décrites précédemment s'appuient notamment sur une approche évolutionniste de la psychologie des émotions, d'autres approches se fondent sur la psychologie environnementale. Ce modèle, développé par Albert Mehrabian [24] qui se fonde sur trois mesures du tempérament, comme décrit plus haut. Ici le tempérament est défini comme la moyenne de ces trois valeurs à travers plusieurs mises en situations représentatives de la vie quotidienne. Selon l'auteur, ces trois mesures sont indépendantes entre elles, et forment le plus petit dénominateur commun aux différentes émotions. Ainsi, ces scores permettent de former des axes orthogonaux pour la représentation des émotions mesurées. Plus spécifiquement on peut décrire les trois axes ainsi :

- L'axe **plaisir/déplaisir** (P) mesure à quel point l'émotion ressentie suscite du plaisir ou non. Ainsi la peur et la colère sont toutes deux associées à une valeur D négative (déplaisir) tandis que la joie est associée à une valeur D positive (plaisir)
- L'axe **éveil/sommeil** (A) montre dans quelles mesures une émotion est énergique. Cette notion n'est pas à confondre avec l'intensité ou l'emphase d'une émotion. Ainsi le chagrin, qui peut être intense, est associé à une mesure A plutôt faible.
- L'axe **dominance/soumission** (D), représente le sentiment de dominance ou de contrôle, ou au contraire de soumission, qui transparaît à travers l'émotion ressentie. De ce fait, si l'anxiété et l'hostilité sont toutes deux des émotions négatives et plutôt énergiques. Mais si la première est associée à une valeur D négative (l'anxiété est subie), la seconde a une valeur positive.

On peut alors constater que des émotions s'opposent en diagonale dans cet espace. Selon l'auteur de l'article présentant cette approche [24], ces diagonales permettent de distinguer des traits de personnalité importants. Cette approche fut l'une des premières à vouloir représenter les émotions en trois dimensions, et une variante plus récente en propose une version revisitée [25].

Toutefois si les valeurs P et A sont utilisées depuis longtemps en psychologie [26], on peut remplacer la troisième valeur (D), pour former un autre modèle. Ainsi, Alexander Lantijak [27] propose de conserver les deux dimensions liées à la perception de l'émotion (P et A), mais de remplacer la valeur de dominance par une échelle de temporalité, suivant si l'émotion ressentie suit ou précède l'événement qui lui est associé. Par exemple, si l'anxiété et la colère sont toutes deux des émotions négatives à forte énergie, l'anxiété fait référence à un événement à venir tandis que la colère a pour objet un événement passé. Ainsi, on obtient, à l'instar du modèle PAD, un espace en trois dimensions sur lequel on peut placer une vingtaine d'émotions comme on peut le voir en figure 3.5.

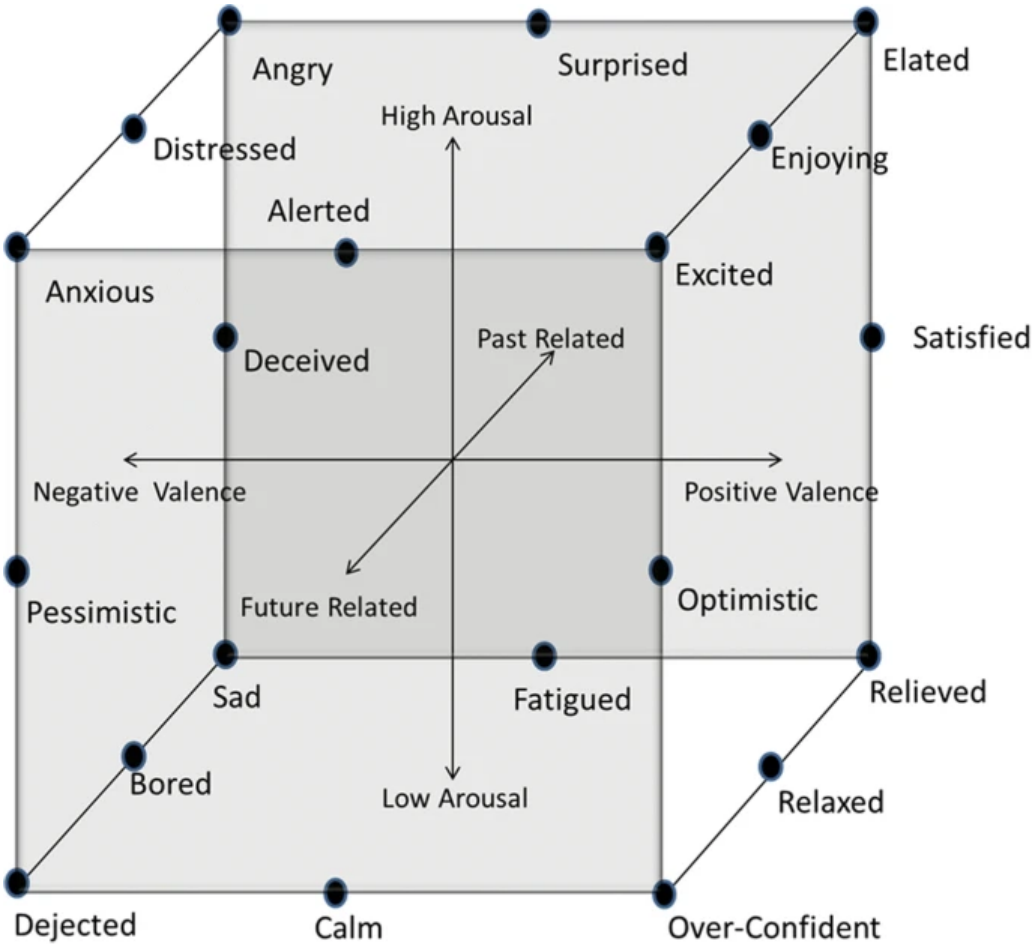


FIGURE 3.5 : Concepts d'émotions au sein du modèle tri-dimensionnel de Latinjak [27]

Le modèle de Plutchnik

En 1982, Robert Plutchnik présente son modèle de représentation des émotions [28]. Ce modèle se structure autour de 8 axes émotionnels formant une roue comme on peut le voir en figure 3.6. On peut remarquer, que les émotions opposées sur la roue se veulent contraires. Chaque axe émotionnel peut alors se décomposer en plusieurs émotions. L'ensemble de la structure forme le patron d'une structure en 3 dimensions[7] comme en figure 3.7.

Une variante plus complexe de cette approche est le modèle dit du "sablier des émotions" (*Hourglass of emotions*), dont la représentation en trois dimensions prend la forme d'un sablier [12, 29], visible en figure 3.8. Ce modèle se veut applicable dans le contexte de l'interaction homme-machine, notamment pour mesurer respectivement si l'utilisateur est amusé par les interactions (Pleasantness), intéressé (Attention), à l'aise avec l'outil (Sensitivity), ou confiant

7. les émotions n'étant pas ici orthogonales

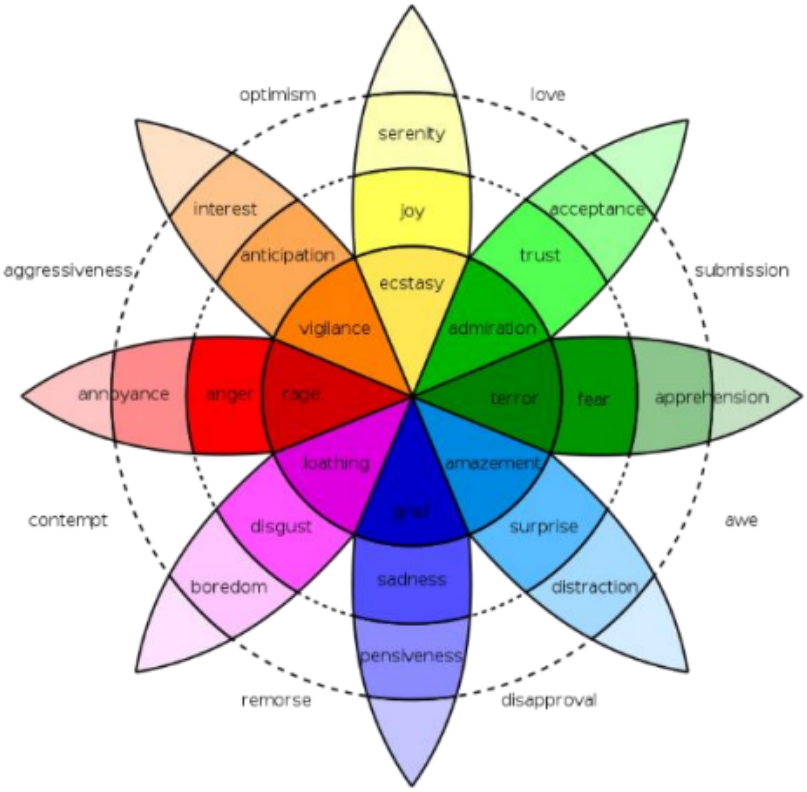


FIGURE 3.6 : Schéma de la roue des émotions de Plutchnik [15]

dans son usage (Aptitude).

3.2.3 Bilan des modèles

Finalement, l'ensemble de ces modèles présente une forte diversité, et l'on peut se figurer une classification comme celle présentée en figure 3.9. Comme nous l'avons vu, chacun de ces modèles possède ses avantages et limites. Cependant, notre choix pour la détection d'émotions est principalement dicté par la disponibilité des ressources. Ainsi, même si d'un point de vue psycholinguistique le modèle d'Ekman est discutable, il est possible de trouver des méthodes s'en servant pour le français. De plus, bon nombre de modèles se fondent sur cette approche, qui constitue une base solide. Ainsi, c'est ce modèle que nous avons retenu pour la suite de notre travail pour la détection d'émotions. En revanche, la détection d'opinion est bien documentée, et il est possible de fournir un complément aux émotions d'Ekman.

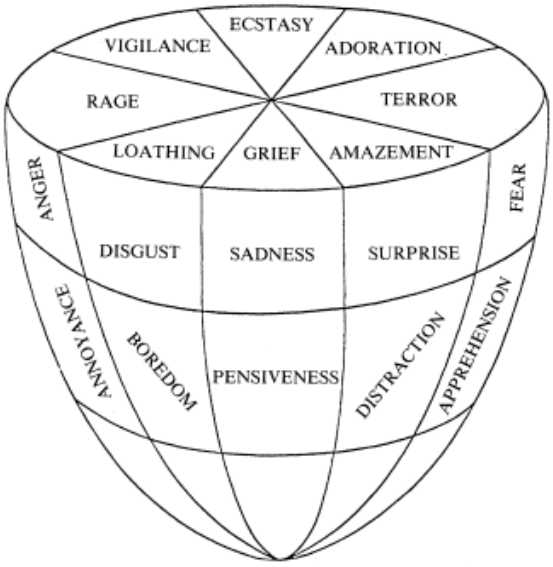


FIGURE 3.7 : Visualisation en 3D du modèle de Plutchnik [28]

3.3 Construction d'un modèle d'analyse d'émotions

Il est probable que les émotions ne soient pas nécessairement le critère le plus influent sur l'impact de la réception d'une *newsletter* et le comportement du receveur. Ainsi un envoi trop intensif de *newsletters* dans un court laps de temps, l'utilisation de certains mots pouvant être considérée comme des *spam words*, ou encore une longueur trop importante, peuvent faire en sorte que la *newsletter* soit perçue comme une nuisance ou un *spam*. Cependant, si l'on constate que les communications par e-mail ont pris beaucoup d'ampleur dans notre quotidien, elles ont aussi tendance à devenir plus informelles et donc à véhiculer davantage d'émotions. Il est donc intéressant de voir dans quelle mesure ces émotions influent sur la performance des campagnes d'e-mail.

Certaines hypothèses proposées par des études de marketing peuvent s'appliquer à notre problème d'impact des émotions dans les communications par *e-mail*. Par exemple, une étude proposée par [30] suggère que le manque d'interactions face à face dû au contexte des communications par *e-mail*, peut donner lieu à une mauvaise interprétation des émotions.

Selon l'auteur, le manque d'indices sur les intentions des émotions de l'envoyeur, peut induire le receveur de l'*e-mail* en erreur. Cette incompréhension menant bien souvent à un *effet de neutralité* ou même à un *effet de négativité*. L'auteur précise que, faute d'indices suffisants sur les émotions venant de l'auteur du mail, les émotions négatives dans le texte seraient plus saillantes. Ainsi un sarcasme, par exemple, pourrait être moins bien perçu par *e-mail* que dans une relation face à face. Cette étude met de plus l'accent sur le contexte social au sein de

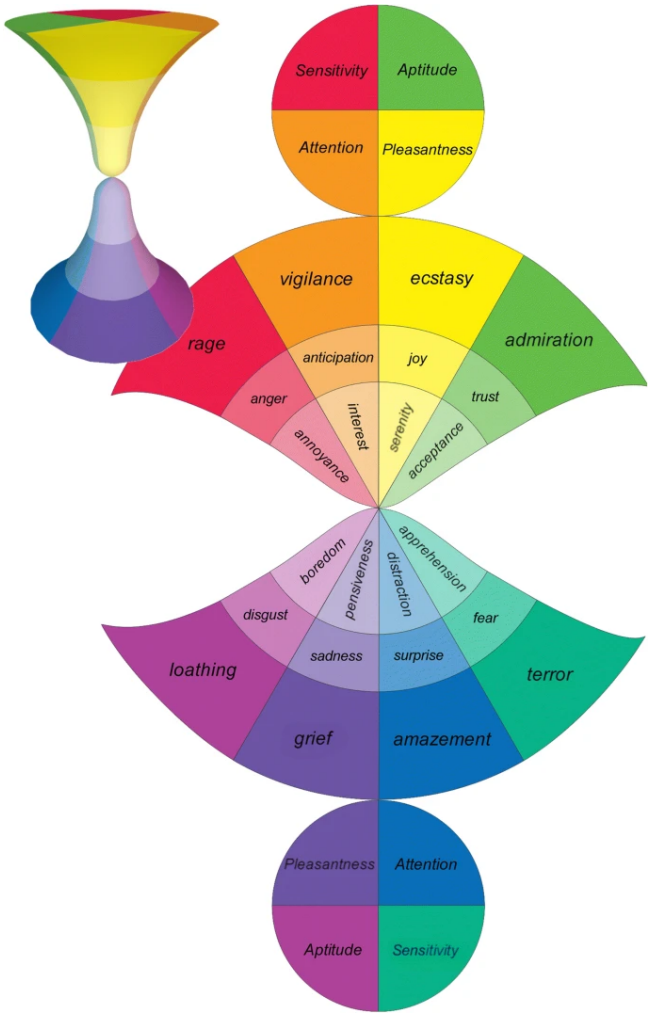


FIGURE 3.8 : Visualisation en 3D du modèle du sablier d'émotions [12]

l'entreprise pour expliquer cet effet (genre, âge, hiérarchie, etc.) Cependant, l'auteur pointe aussi des conséquences positives à cet effet, comme la non-nécessité de chercher à embellir les *e-mails* d'émotions positives qui ne seraient pas, ou mal perçues. Ce biais de négativité pourrait donc poser problème dans un contexte de *newsletters*, pour lesquels transmettre des émotions positives comme la fierté [31] semble crucial pour obtenir des attitudes positives de la part des consommateurs, et plus particulièrement dans les cultures occidentales.

De plus, le fait que, lors des campagnes d'*e-mail*, envoyeur et receveur ne se connaissent pas, accentue encore davantage la méconnaissance de certains éléments du contexte et peut, par conséquent, mener d'autant plus facilement vers une mauvaise interprétation des émotions. Enfin, une étude récente de [32] portant sur les conduites de *newsletters* pendant la pandémie du COVID-19 tend à montrer que les communications commerciales par *e-mail* ne sont plus

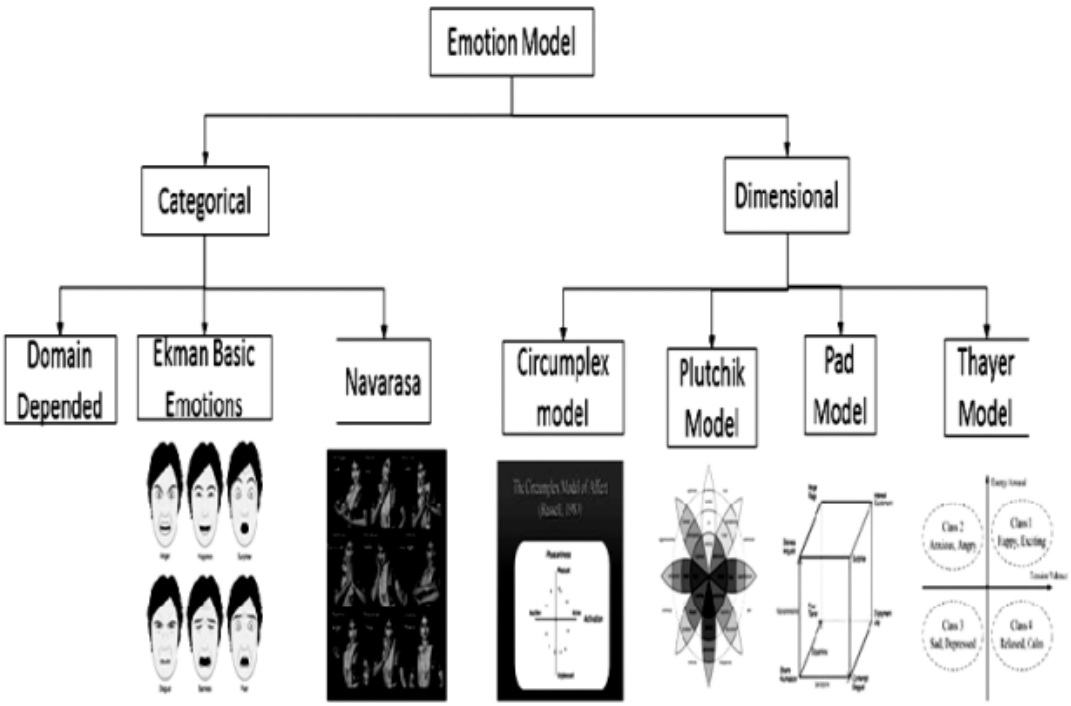


FIGURE 3.9 : Arborescence des principales méthodes de détection d'émotions dans le texte [14].

simplement informatives mais qu'elles contiennent également aujourd'hui des éléments divertissants et avec un ton moins formel. Cela peut se traduire par des newsletters plus personnelles, et moins uniquement informatives. Or, ce type de discours aura plus tendance à faire intervenir des émotions pour susciter de l'intérêt. Tout ceci fait que l'attention portée à l'expression des émotions devient un élément essentiel dans la communication des entreprises pour éviter d'éventuelles erreurs d'interprétation.

L'impact des émotions sur les performances de campagne d'*e-mails* a été étudié par R. Miller et E. Charles, [33], qui ont validé plusieurs hypothèses sur la manière dont les émotions issues de l'objet d'un mail peuvent influencer sa perception. Les auteurs ont travaillé sur le jeu de données de Enron ([34]), composé d'*e-mails* rédigés en anglais et issus de communications internes à une même entreprise.

3.3.1 Approche par lexique

L'une des méthodes les plus simples pour détecter les émotions est d'utiliser un lexique de mots ayant ou non une connotation avec chacune d'elles. À ce jour, l'un des lexiques les plus complets en langue française est celui de [35], obtenu par traduction du lexique NRC-EmoLex de l'anglais vers le français. Ce lexique a ensuite été enrichi et validé par des traducteurs professionnels, pour

un total de plus de 14000 lemmes ; chaque lemme possédant les émotions contenues dans ses formes fléchies. L'ensemble est un ensemble d'environ 12000 lemmes uniques, et 2.148 formules composées. Grâce à ce jeu de données, il est possible de réaliser une analyse des émotions par sacs de mots.

Extraction de caractéristiques

Nous avons représenté le contenu textuel de notre jeu de newsletters en passant par un vecteur de caractéristiques, ou *features*. Le taux d'ouverture et le taux de cliques sont directement obtenus depuis les services hôtes, et forment les indicateurs de performance décrits plus haut. Pour obtenir le reste des informations nous avons utilisé des techniques habituelles du traitement automatique du langage. Tout d'abord nous avons segmenté le texte en phrases (on parle alors de tokenisation), puis lemmatisé. Nous avons ensuite retiré les *stop-words* ou mots-vides, suivant une liste issue de la bilbiothèque python *Spacy*[8] , et affecté une émotion en suivant le lexique FEEL [35]. Si le mot est une forme fléchie, nous considérons alors le lemme correspondant. À ce vecteur représentant les six émotions d'Ekman, nous avons ajouté deux valeurs de polarité et de subjectivité pour l'analyse d'opinion. Enfin le vecteur représentant chaque phrase est déterminé en faisant la moyenne des vecteurs de chaque lemme la composant. Le processus complet est visible en figure 3.10.

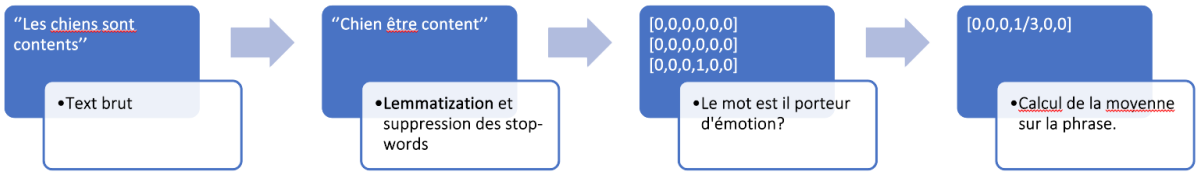


FIGURE 3.10 : Schéma présentant le processus de détection d'émotions par sac de mots

Résultats statistiques

L'approche lexicale nous a montré que la détection d'émotions sur l'objet du mail n'est pas vraiment pertinente, compte tenu du peu de mots qu'il contient. Ainsi concernant l'objet du mail, nous nous sommes simplement intéressés à la polarité et la subjectivité, obtenues via l'outil *Textblob*. La bibliothèque *Textblob* du langage *Python* est un outil gratuit de traitement automatique du langage, qui peut être utilisé afin de de déterminer la polarité et la subjectivité d'un texte court comme une phrase, ou un tweet. Cette méthode, [36], nous permet donc de rajouter ces deux features à notre analyse de texte.

8. `https://github.com/explosion/spaCy/blob/master/spacy/lang/fr/stop_words.py`

Enfin, nous avons ajouté des features non liées à l'analyse d'émotions telles que la longueur de la phrase. La moyenne de tous les vecteurs des phrases forme la représentation du document entier.

Puis nous avons exploré nos données en comparant les corrélations de Pearson entre les features d'émotions et d'opinion et les indicateurs de performance (taux de clique, et taux d'ouverture). Nous avons aussi calculé les corrélations entre ces indicateurs et d'autres features utilisées dans d'autres études [37], telles que la taille du fichier *.eml* d'origine. Les résultats sont présentés en figure 3.1.

TABLE 3.1 : Corrélation de Pearson entre les features et les indicateurs de performance.

Features	**Taux d'ouverture**	**Taux de clique**
Subject line's polarity	(-0.071, 0.026)**	(-0.033, 0.0029)
Subject line's subjectivity	(-0.01,0.75)	(-0.07, 0.028)*
Subject line's length	**(-0.13, 3.75e-5)*** **	**(0.18, 8.6e-9)*** **
File size	**(-0.14, 4.70e-6)*** **	**(0.25, 2.41e-15)*** **
Content Polarity	-	(0.090, 0.0049)**
Content Subjectivity	-	(-0.070, 0.030)*
Content Joy	-	(-0.10, 0.011)**
Content Fear	-	**(-0.11, 0.0008)*** **
Content Sadness	-	**(-0.233, 1.8e-13)*** **
Content Anger	-	(0.056, 0.08)
Content Surprise	-	**(-0.11, 5.5e-4)*** **
Content Disgust	-	**(-0.073, 0.022)***

*p-value $< .05$
**p-value $< .01$
***p-value $< .001$

Ces résultats nous donnent des indices sur l'impact de chaque feature sur la performance des newsletters estimée en taux de cliques et d'ouverture. Tout d'abord, il semble clair que des descripteurs plus classiques tels que la longueur de l'objet du mail ou la taille du fichier sont fortement corrélés avec les performances. Ainsi, plus l'objet du mail est long, ou plus son fichier est lourd, et moins la newsletter envoyée aura d'impact. Cependant, les résultats les plus intéressants concernent les features qui décrivent les émotions. Il apparaît que les émotions venant du contenu de l'email sont corrélées négativement avec le taux de clique, et ce, indépendamment du type de ces émotions. Si ces features émotionnelles ne sont peut-être pas les meilleures pour prédire la performance d'une newsletter, ces corrélations nous incitent à poursuivre l'exploration de notre jeu de données représenté par ces features.

3.3.2 Classification non supervisée

Afin d'explorer plus avant notre jeu de données de newsletters représentées par des émotions, nous avons commencé par nous orienter sur des méthodes de classification non supervisées, afin de déterminer s'il existe une structure globale permettant d'interpréter les newsletters. Cependant, celles-ci sont représentées par un vecteur de dimension 10 (6 features d'analyse d'émotion du contenu, 2 d'analyse d'émotion du contenu, et 2 d'analyse d'émotion de l'objet) ; afin de représenter nos données dans un graphe en deux dimensions, nous avons donc dû réduire le nombre de dimensions. Pour cela nous avons utilisé une t-SNE[9], qui permet de préserver au mieux dans la visualisation les voisinages entre les points. Le résultat de cette projection est en figure 3.11

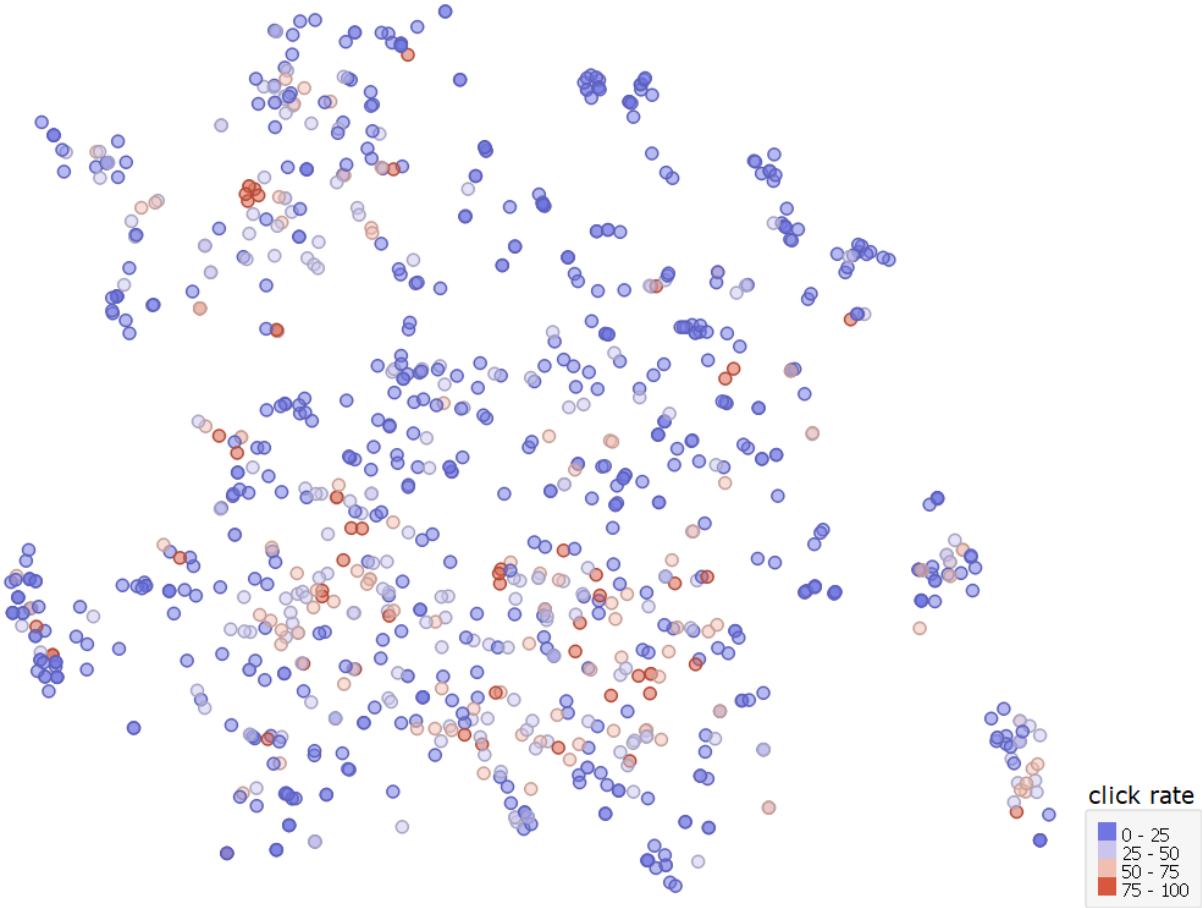


FIGURE 3.11 : Projection via t-SNE de notre jeu de données

À première vue, il semble que nos features ne permettent pas d'en déduire des clusters bien

9. t-distributed stochastic neighbor embedding

distincts : les campagnes de newsletters avec un taux de cliques faible sont réparties uniformément dans tout l'espace. Cependant, il semble aussi que les "bonnes" campagnes, avec un fort taux de clique, sont plus centrées et regroupées. Ainsi, notre première idée fut de regrouper ces bonnes newsletters en une catégorie à l'aide d'algorithmes de classification non supervisée.

Approche par k-moyennes

L'algorithme *k-means* (ou algorithme des k-moyenne), est une méthode de clustering qui vise à partionner notre jeu de données en k clusters dans lesquels chaque point ou donnée, est associée à un point moyen (centre du cluster le plus proche). Or, nous ne savons pas *a priori* combien de clusters sont nécessaires pour obtenir un partitionnement pertinent de nos données. Afin de déterminer le nombre de clusters, nous avons procédé à une ACP (analyse en composantes principales) [10], afin de réduire le bruit dans nos données, puis nous avons fait varier le nombre de clusters utilisés par l'approche par k-means, pour ne garder que le nombre minimum de clusters qui maximise l'explication de la variance. Le but ici, est de trouver le nombre de clusters qui maximise le score de silhouette des clusters.

Le score de silhouette indique si les points du cluster sont dispersés ou concentrés autour de son centre. Ainsi, pour k clusters obtenus après l'application de l'algorithme de k-moyennes, soit un point $i \in I_k$ (data point i dans le cluster I_k), on définit :

$$a(i) = \frac{1}{|I_k| - 1} \sum_{j \in I_k, j \neq i} d(x^i, x^j) \tag{3.1}$$

Avec $d(x^i, x^j)$ la distance entre les points i et j. Dans notre cas, à cause du grand nombre de dimensions de notre espace, nous avons choisi la fonction *cosine* comme fonction de similarité d. La valeur $a(i)$ (3.1) représente à quel point un point (i) est proche des autres points du même cluster. De la même manière, nous pouvons définir la valeur $b(i)$ qui représente si le même point est proche des autres points des autres cluster les plus proches (3.2).

$$b(i) = \min_{k' \neq k} \frac{1}{|I_{k'}| - 1} \sum_{i' \in I_{k'}} d(x^i, x^{i'}) \tag{3.2}$$

On peut alors calculer le score de silhouette d'un point i comme suit (3.3) :

$$\begin{aligned} s_{silhouette}(i) &= \frac{b(i) - a(i)}{\max\left(a(i), b(i)\right)} \quad if : |C_i| > 1, \\ s_{silhouette}(i) &= 0 \qquad\qquad\qquad\qquad if : |C_i| = 1 \end{aligned} \tag{3.3}$$

10. Principal Component Analysis (PCA)

On peut alors calculer le nombre de clusters avec le meilleur score de silhouette pour chaque configuration avec un nombre de composantes principales différent. Les résultats sont montrés en table 3.2.

TABLE 3.2 : Nombre de clusters avec le meilleur score de silhouette, selon le nombre de composantes principales gardées.

PCA	**Variance expliquée**	**Nombre de clusters**[a]	**Score de silhouette**
1	24%	2	0.577
2	40%	2	0.501
3	53%	4	0.411
4	63%	2	0.358
5	72%	2	0.274
6	79%	2	0.269
7	86%	3	0.250
8	91%	2	0.258
9	96%	4	0.392
10	100%	4	0.366

[a] Le nombre de clusters pris en compte est celui qui maximise le score de silhouette.

Il apparaît que deux clusters semblent être la configuration qui permet de séparer notre jeu de données de la manière la plus cohérente d'après le score de silhouette. De plus, 8 composantes principales semblent être suffisantes pour expliquer plus de 91% de la variance des données de notre jeu de données, c'est pourquoi nous avons gardé cette configuration pour la suite de notre analyse. Pour déterminer si ces deux clusters ainsi obtenus, censés séparer les "bonnes" et "mauvaises" newsletters, permettent effectivement de mesurer les performances des newsletters, nous avons analysé la distribution du taux de cliques dans chacun des clusters. Ces distributions sont montrées en figure 3.12.

Ces résultats nous indiquent que les clusters obtenus par k-moyennes ne sont pas représentatifs de la distribution du taux de cliques. Dans la prochaine partie, nous adoptons une autre approche qui s'appuie sur une classification supervisée.

3.3.3 Classes de performance

La mise en œuvre d'une classification supervisée nécessite une labellisation préalable de nos données. Pour ce faire, nous avons séparé notre jeu de données en catégories suivant le taux de cliques. Nous avons divisé notre jeu de données en deux moitiés égales, pour former deux clusters. La première concerne les "mauvaises" newsletters, soit les 50% ayant le taux de cliques

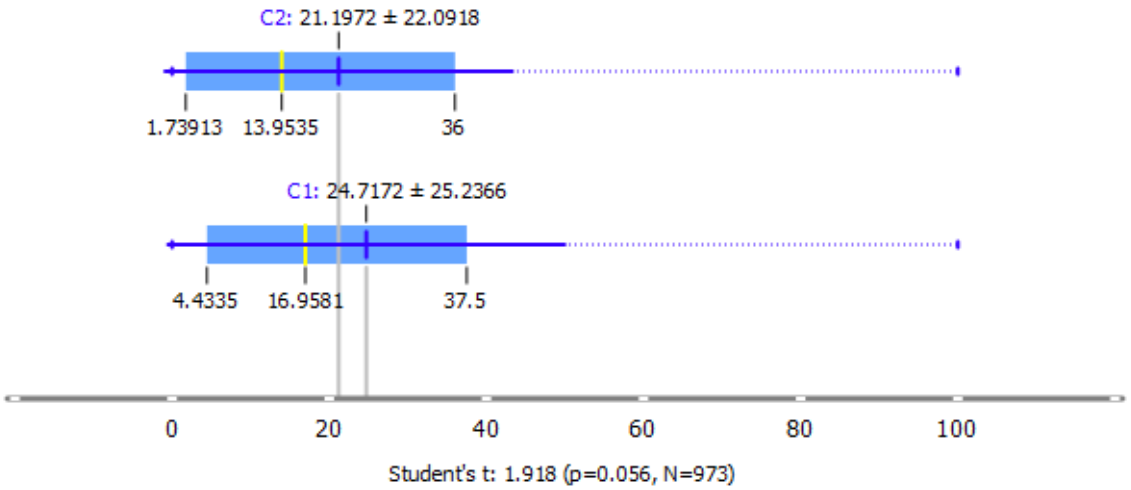


FIGURE 3.12 : Diagramme en boîtes à moustaches comparant les taux de cliques pour chacun des deux clusters obtenus par k-moyennes, avec 8 composantes principales. Le test de Student nous indique que ces deux clusters ne sont pas statistiquement différenciables par le taux de clique. Ce test fut effectué sur un jeu de donnée de taille N=973, comprenant les données filtrées par la suite (doublons, etc). Les conclusions sur le jeu de 799 newsletters sont les mêmes

le plus faible, et la seconde est constituée des 50% ayant le taux de cliques le plus élevé. La médiane du taux de cliques forme ainsi le séparateur de notre jeu de données. Nous pouvons alors mesurer le score de silhouette de chacun des deux clusters ainsi obtenus, nous pouvons voir ces résultats en figure 3.13.

On peut observer que les newsletters avec un taux de cliques faible ont un score de silhouette assez bas, et que celle avec un taux de cliques élevé ont un score de silhouette plus élevé. Cela confirme donc l'intuition évoquée précédemment, à savoir que si les bonnes newsletters forment un groupe cohérent resserré, les mauvaises sont réparties plus uniformément dans notre espace. De plus, on peut voir en figure 3.14 que cet effet est encore plus fort lorsque l'on retire les informations liées à l'objet du mail (polarité et subjectivité).

Si nous ne pouvons conclure de manière tranchée, ces résultats tendent à montrer qu'il est possible de séparer notre jeu de données en ces deux clusters et donc qu'il est possible d'appliquer ces labels à nos données en vue d'une tâche de classification supervisée.

3.3.4 Classification supervisée

En suivant la perspective mentionnée précédemment, nous avons donc associé un label à nos données suivant deux classes : les "bonnes" newsletters, qui sont au dessus de la médiane du taux de clique, et les "mauvaises", qui sont en-dessous. Puisque nous ne nous intéressons ici qu'à deux classes, nos métriques pour la classifications seront celles de classification binaires,

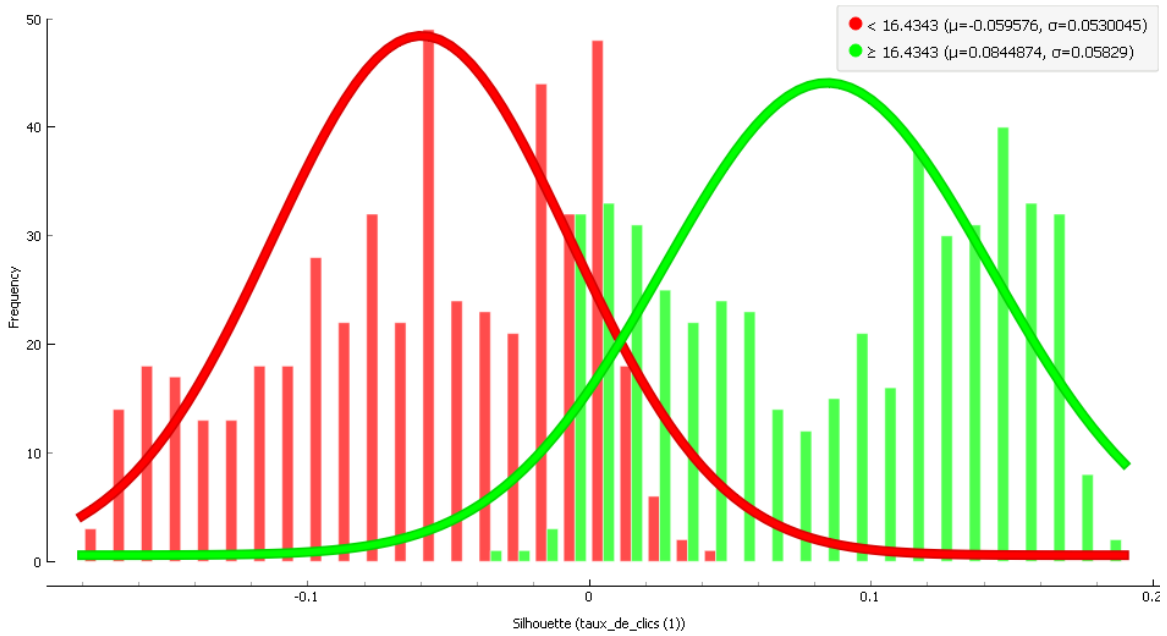


FIGURE 3.13 : Distribution du score de silhouette pour les deux clusters suivant le taux de cliques bas (en rouge) ou élevé (en vert).

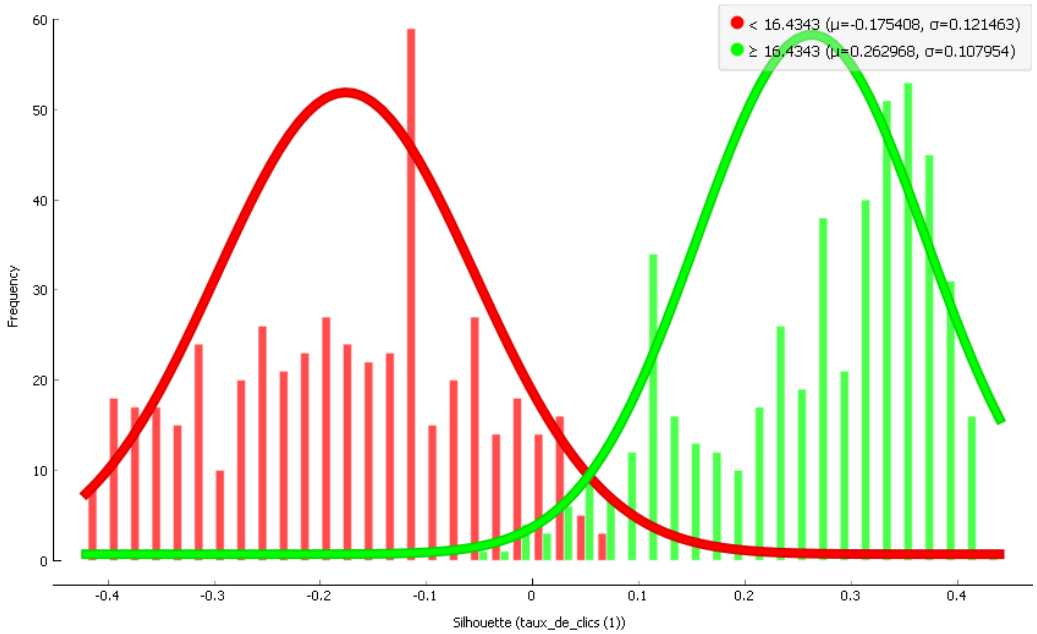


FIGURE 3.14 : Distribution du score de silhouette pour les deux clusters suivant le taux de cliques bas (en rouge) ou élevé (en vert) ; mais sans les features issues de l'analyse de l'objet du mail.

en considérant les mauvaises newsletters comme la classe "0". Notre tâche consiste donc ici à prédire ces classes, en utilisant notre *embedding*, ou plongement d'émotions et d'opinions. Pour cette tâche, nous avons pris en compte les features décrites précédemment. Pour ce faire, nous avons testé, plusieurs algorithmes souvent utilisés en machine learning [38]. Les résultats de cette tâche de classification sont présentés en table 3.3, où chaque prédiction a été obtenue après une procédure de validation croisée avec 10 plis appliquée au jeu de données.

TABLE 3.3 : F1-score, précision et rappel, de la tâche de classification suivant les modèles, et avec ou sans les features issues de l'objet du mail

Modèle	**F1 Score**	**Précision**	**Rappel**
AdaBoost	**0.723**	**0.724**	**0.724**
Réseaux de Neurones	0.712	0.712	0.712
Forêts aléatoires	0.711	0.711	0.711
kNN	0.681	0.688	0.683
Naive Bayes	0.666	0.666	0.666
SVM	0.607	0.617	0.612
Régression logistique	0.585	0.594	0.590
Constant	0.500	0.500	0.500
Résultats avec uniquement les informations sur le contenu du mail			
Modèle	**F1 Score**	**Précision**	**Rappel**
AdaBoost	**0.722**	**0.723**	**0.723**
Réseaux de neurones	0.714	0.715	0.715
Forêt aléatoires	0.710	0.710	0.710
kNN	0.679	0.683	0.680
Naive Bayes	0.666	0.666	0.666
SVM	0.628	0.640	0.633
Régression logistique	0.621	0.643	0.630
Constant	0.500	0.500	0.500

Si nous pouvons espérer de meilleurs résultats pour chaque méthode, en ajustant leurs hyper-paramètres, il apparaît que les meilleurs méthodes sont l'algorithme AdaBoost, les réseaux de neurones, et les forêts aléatoires. Nous pouvons aussi remarquer que ces scores sont légèrement moins efficaces sans les features issues de l'objet du mail. Afin de déterminer quelles sont les features ayant le plus d'impact sur cette prédiction, nous avons choisi notre meilleur modèle (ici AdaBoost), et observé comment les résultats de la prédiction pouvaient varier selon que l'on considère une unique feature en entrée, ou bien toutes les features sauf une en particulier. Nous avons appliqué cette méthode dîtes *Ablation study* pour chacune de nos 10 features d'origine, les résultats sont présentés en table 3.4.

Ces résultats tendent à confirmer l'impact des features émotionnelles vu précédemment. La

TABLE 3.4 : Résultats de la classification avec seulement une feature, ou bien toutes sauf celle testée.

Feature testée	**F1-score avec uniquement la feature testée**	**F1-score avec toutes les features sauf celle testée**
Polarité de l'objet du mail	0.498	0.720
Subjectivité de l'objet du mail	0.503	0.721
Polarité du contenu	0.614	0.719
Subjectivité du contenu	0.570	0.725
Joie dans le contenu	0.624	0.723
Peur dans le contenu	0.604	0.722
Tristesse dans le contenu	**0.633**	**0.711**
Colère dans le contenu	0.618	0.713
Surprise dans le contenu	0.614	0.721
Dégoût dans le contenu	0.626	0.721

tristesse étant l'émotion ayant le plus grand impact sur les performances. Cependant notre approche par lexique a ses limites. En effet elle ne permet pas de déterminer la charge émotionnelle dans une tournure de phrase donnée, où aucun mot n'est associé à une émotion selon le lexique. Cette lacune peut cependant être comblée en utilisant une autre approche, faisant ici place à l'apprentissage profond (*deep learning*).

3.3.5 Approche par modèle entraîné à détecter les émotions

Comme nous l'avons vu, faute de prendre en compte le contexte du mot au sein d'une phrase, l'approche par lexique peut mener à de mauvaises interprétations, voire à des contresens. Une autre approche consiste à entraîner un modèle pour attribuer automatiquement une émotion à une phrase. Par exemple, en utilisant un modèle auto-encodeur multilingue, on peut, par ré-apprentissage, l'entraîner à détecter automatiquement si une phrase donnée renvoie à une certaine émotion.

Dans cet esprit, nous avons traduit en français le jeu de données proposé par Saravia et al [39] constitué de phrases en anglais et des émotions qui leur sont attribuées. Les auteurs ont généré un jeu de données étiqueté en utilisant un algorithme semi-supervisé basé sur une approche par graphes enrichie à l'aide d'un plongement de mots. Leur méthode d'enrichissement de données semble très performante par rapport à l'état de l'art, et leur jeu de données comporte 20 000 phrases associées chacune à une émotion. Les auteurs suggèrent d'utiliser ce jeu de données pour

ré-entraîner un auto-encodeur d'architecture T5 [40].

Dans leurs travaux, Saravia et al, [39] associent à des phrases en anglais une émotion parmi les 6 émotions d'Ekman décrites précédemment, à l'exception du *dégoût*, et avec l'*amour* comme nouvelle catégorie. Nous avons traduit ces phrases en français à l'aide d'un traducteur automatique pour ensuite réaliser un ré-apprentissage. Pour ce faire, nous avons utilisé *Google translate* dont une étude récente, [41], relève les bonnes performances pour la traduction de l'anglais vers le français. Par ailleurs, notre apprentissage se fait sur des données issues de tweets, donc assez courtes, pour espérer que la traduction n'engendre pas trop d'erreurs. Les performances du modèle ainsi obtenu figurent dans le tableau 3.5.

Cette méthode est similaire à celle utilisée dans le chapitre 1, et consiste à adapter un modèle appris sur de l'anglais à du français via un apprentissage par transfert. Un schéma de cette application est en figure 3.15.

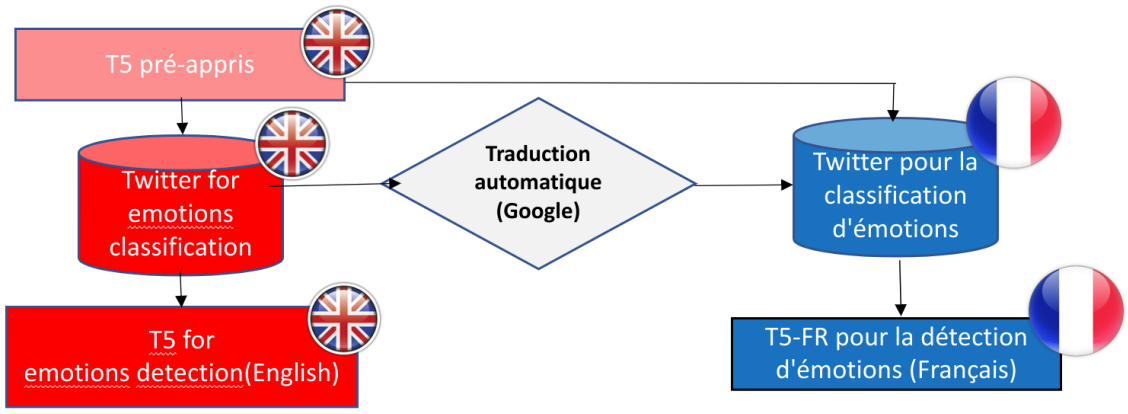


FIGURE 3.15 : Schéma représentant la méthode d'apprentissage transfert, visant à adapter le modèle T5 de détection d'émotion depuis l'anglais vers le français.

On observe que la précision du modèle chute en moyenne d'environ 20% par rapport à la référence du modèle originel en anglais, avec des différences notables en fonction des émotions. Ainsi, si le modèle perd jusqu'à près de 50% de ses performances pour détecter l'émotion *amour*, la perte n'est que de 5% pour la *surprise*. On peut voir plus en détail les résultats des tests de prédiction avec la matrice de confusion donnée en figure 3.16.

Il apparaît, que si effectivement la prédiction de l'émotion *amour* est mauvaise, cela est surtout dû à la confusion du modèle avec la *joie* et, dans une moindre mesure, la *tristesse*. Sans faire de conclusions trop hâtives sur l'interprétation des émotions par le modèle, on peut toutefois remarquer que *joie* et *amour* sont les deux seules émotions *positives* de l'ensemble du panel d'émotions.

TABLE 3.5 : Résultat du modèle T5 à l'issue d'un ré-apprentissage (*fine tuning*) visant à prédire des émotions sur des données traduites en français, et la différence de performance par rapport au modèle originel en anglais.

Emotion	**Précision**	**Rappel**	**F1-score**	**Support**
Colère	0,72 *(-22%)*	0,64 *(-29%)*	0,68 *(-25%)*	275
Peur	0,73 *(-13%)*	0,66 *(-26%)*	0,69 *(-20%)*	224
Joie	0,80 *(-17%)*	0,85 *(-8%)*	0,83*(-12%)*	695
Amour	0,55 *(-24%)*	0,29 *(-60%)*	0,38*(-46%)*	159
Tristesse	0,74 *(-23%)*	0,84*(-12%)*	0,79*(-18%)*	581
Surprise	0,69*(-6%)*	0,71*(-3%)*	0,70*(-5%)*	66
Exactitude			0,75*(-18%)*	2000
Moyenne	0,70*(-18%)*	0,67*(-13%)*	0,68*(-21%)*	2000
Moyenne pondérée	0,74*(-19%)*	0,75*(-18%)*	0,74*(-19%)*	2000

Comparaison des méthodes de prédiction de performance des newsletters

Tout d'abord on peut voir que, comme voulu, le modèle T5 ainsi obtenu permet d'associer une émotion là où l'approche par sac de mots ne le permet pas, comme le montre l'exemple présenté en figure 3.17.

On peut remarquer certaines particularités observées lors de l'application du modèle T5 au jeu de données. On constate que la joie est l'émotion la plus souvent détectée, certaines *newsletters* ayant jusqu'à 80% de leur phrases labellisées comme "joyeuses". Bien que la joie puisse être le ton par défaut dans une communication commerciale, cette sur-représentation de la joie est imputable à l'absence d'un label. Le label "neutre" est très souvent le label ayant la plus forte occurrence dans les études sur la détection des émotions où il figure.

À l'inverse, dans l'ensemble de notre jeu de données, aucune phrase n'est labellisée avec l'émotion *colère* ; ce qui peut aisément se concevoir dans un contexte de communication commerciale.

On peut aussi remarquer que la détection de *amour* est corrélée négativement avec le taux de cliques. Cependant, le modèle étant peu précis pour la détection de cette émotion en particulier, nous ne conclurons pas sur l'impact du ton employé par l'auteur de l'*e-mail*.

Enfin, et surtout, on peut remarquer que, hormis la joie détectée par T5, les émotions sont négativement associées aux taux de cliques générés. Les corrélations entre les différentes caractéristiques extraites et les indicateurs de performance figurent dans le tableau 3.6.

D'une manière générale, les caractéristiques issues du modèle T5 entraîné sur le français ont une corrélation moindre avec le taux de cliques que les caractéristiques issues de l'approche par sac de mots issue du lexique *FEEL*. Ces deux approches étant complémentaires nous avons donc testé plusieurs combinaisons de features.

FIGURE 3.16 : Matrice de confusion des prédictions du modèle T5 ré-entraînés pour la détection d'émotions en français.

Approche par sac de mots :
"*Je tiens à toi* " -> [Je][tenir][toi]-> [0^{em},0^{em},0^{em}]-> **Aucune Emotion détectée**

Approche par modèle T5 entrainé :
"*Je tiens à toi* " -> **Amour**

FIGURE 3.17 : Exemple de phrase traitée par les deux approches. Ici 0^{em} désigne le vecteur nul pour l'espace d'émotions considérées. On peut voir que l'approche par sacs de mots ne permet pas de labéliser correctement les émotions du fait de l'absence de marqueur lexical d'émotion. En revanche le modèle T5 adapté au français y parvient.

TABLE 3.6 : Corrélation de Pearson entre les caractéristiques des *newsletters* et leurs performances, suivant les deux approches : par lexique (en noir) et avec le modèle T5 réappris (en bleu)

Caractéristique	**Taux d'ouverture**	**taux de cliques**	
Taille du fichier	**-0.14*** **	**0.25*** **	
Longueur de l'objet	**-0.13*** **	**0.18*** **	
Polarité de l'objet	-0.07**	-0.03^{ns}	
Subjectivité de l'objet	-0.01^{ns}	-0.07*	
Polarité du contenu	-	0.09**	
Subjectivité du contenu	-	-0.07*	
Joie dans le contenu	-	-0.10**	0.06*
Peur dans le contenu	-	**-0.11*** **	-0.02^{ns}
Tristesse dans le contenu	-	**-0.23*** **	-0.06^{ns}
Colère dans le contenu	-	0.06^{ns}	-
Surprise dans le contenu	-	**-0.11*** **	-0.01^{ns}
Dégoût/Amour dans le contenu	-	-0.07*	**-0.12*** **

*p-value $<$ .05, **p-value $<$ .01, ***p-value $<$ .001,$^{\text{ns}}$ not significant

Le tableau 3.7 présente les mesures de performance estimées par la procédure de validation croisée portant sur un partitionnement en 10 sous-ensembles. Les différents types de caractéristiques considérées dans nos différents plongements sont listés ci-dessous :

1. **FEEL** - 6 émotions : joie, peur, colère, surprise, tristesse et dégoût ; issues de notre approche par lexique calculées sur l'ensemble du contenu du mail.

2. **T5_Fr** - 6 émotions : joie, peur, colère, surprise, tristesse et amour ; issues de notre approche utilisant un modèle réappris pour la détection d'émotions en français, et appliqué sur l'ensemble du contenu du mail.

3. **Subj&Pol** : score de polarité et subjectivité calculé sur l'ensemble du mail via une méthode automatisée issue d'un modèle entraîné pour l'analyse d'opinion obtenus comme vu précédemment.

4. **Objet** : score de polarité et subjectivité calculé sur l'objet du mail uniquement via une méthode automatisée issue d'un modèle entraîné pour l'analyse d'opinion.

Ainsi, il apparaît que les méthodes *AdaBoost*, *réseaux de neurones* et *forêts aléatoires* sont les plus performantes pour notre tâche. De plus, bien que plus naïve, l'approche par sac de mots et lexique d'émotions permet d'obtenir une représentation d'émotions plus pertinente pour notre tâche que celle obtenue via le modèle pré-entraîné. On peut arguer que le passage par la

TABLE 3.7 : F1-Score des différents classifieurs exploitant les différents plongements.

Embedding	AdaBoost	RNN	Rand. Forest	kNN	Naive Bayes	SVM	Rég Log
FEEL	0,718	0,702	0,711	0,690	0,656	0,621	0,643
FEEL+Subj&Pol	0,721	0,711	0,711	0,679	0,666	0,621	0,628
FEEL+Subj&Pol+Objet	0,724	0,716	0,713	0,680	0,666	0,607	0,585
T5_Fr	0,697	0,651	0,689	0,676	0,641	0,620	0,616
T5_Fr+Subj&Pol	0,697	0,651	0,689	0,676	0,641	0,620	0,616
T5_Fr+Subj&Pol+Objet	0,693	0,656	0,687	0,657	0,642	0,609	0,610
FEEL+T5_Fr	0,732	0,704	0,722	0,693	0,657	0,651	0,615
FEEL+T5_Fr+Subj&Pol	**0,737**	0,712	0,720	0,702	0,666	0,630	0,629
FEEL+T5_Fr+Subj&Pol+Objet	0,733	0,713	0,721	0,686	0,664	0,637	0,623

traduction augmente le taux d'erreur du modèle T5 en français en produisant des *embeddings* moins précis et donc moins pertinents ; ce qui vient s'ajouter au fait que le modèle réalise des prédictions discrètes (0 ou 1) et associe une émotion unique à chaque phrase. Il aurait sans doute été plus efficace de prédire pour chaque émotion un indice de confiance compris entre 0 et 1 et par suite, d'assigner plusieurs émotions à une même phrase.

Cependant, on peut aussi noter une bonne amélioration en fusionnant les deux approches pour obtenir, par simple concaténation, des plongements qui cumulent les informations apportées par chacune d'elles. En revanche, si le cumul des représentations d'émotions améliore la qualité des prédictions, la prise en compte de la subjectivité et de la polarité de l'objet du mail semble la réduire.

3.4 Conclusions et perspectives

Nous avons étudié l'apport d'une détection automatique d'émotions et d'opinions dans le but d'aider à la prédiction des performances de *newsletters* en langue française. En effet, la littérature dans le domaine des études de marketing suggère que la communication par *e-mail* tend à renforcer les mauvaises interprétations des émotions, à cause d'un biais de négativité ou de neutralité. Lorsque le receveur du mail est un potentiel client ou un abonné, cet effet négatif peut être observé par le biais de mesures objectives dont nous disposons dans notre jeu de données, telles que le taux d'ouverture ou le taux de cliques généré par la *newsletter*.

Notre choix de modèle d'émotions a surtout été dicté par la disponibilité des ressources ainsi que par la comparaison possible avec les résultats indiqués dans la littérature. C'est pourquoi, même si le modèle catégoriel des six émotions d'Ekman possède des faiblesses, c'est celui que nous avons retenu pour la suite. À ces émotions nous avons toutefois ajouté des features d'analyse d'opinion, qui sont moins catégorielles et peuvent être vues comme complémentaires aux émotions d'Ekman. Nous associons aux newsletters un *embedding* d'émotions selon différentes méthodes. Le premier repose sur l'utilisation d'un lexique associant des lemmes à des émotions,

en exploitant une méthode par sac de mots. L'autre est basé sur un auto-encodeur réappris afin d'assigner une émotion à une phrase en français. Cette première étude montre que la plupart des émotions sont corrélées négativement avec les performances des *newsletters*, ce qui vient confirmer l'hypothèse selon laquelle les émotions, qu'elles soient positives ou négatives, tendent à être interprétées négativement.

Ensuite, nous avons utilisé des *embeddings* d'émotions et de sentiments pour prédire les performances des *newsletters*. Nous avons présenté différentes approches qui se différencient par la nature de l'*embedding* exploité. Il ressort de notre étude qu'une détection d'émotions combinant une approche par sac de mots et par un modèle ré-appris, permet de prédire les classes de performances plus efficacement que chacune de ces deux méthodes prises séparément. En intégrant des approches plus poussées pour la caractérisation d'émotions, nous espérons encore améliorer la prédiction de performance des *newsletters*. Ainsi une perspective future d'amélioration serait de choisir des émotions ou des méthodes de détections d'émotions qui soient plus en rapport avec notre contexte.

Enfin, toujours dans l'optique d'aider à l'édition de newsletters, notre approche ne prend pas assez en compte les caractéristiques propres de ces campagnes de mails. Actuellement, il est difficile d'appréhender les conclusions du modèle en vue d'une correction du ton employé. Ainsi prendre en compte tout le texte en un seul bloc et uniquement les émotions n'est pas suffisant.

BIBLIOGRAPHIE

[1] P. Nandwani and R. Verma, "A review on sentiment analysis and emotion detection from text," Social Network Analysis and Mining, vol. 11, no. 1, pp. 1–19, 2021.

[2] S. Bird, E. Klein, and E. Loper, Natural language processing with Python : analyzing text with the natural language toolkit. " O'Reilly Media, Inc.", 2009.

[3] S. Loria et al., "textblob documentation," Release 0.15, vol. 2, no. 8, 2018.

[4] T. De Smedt and W. Daelemans, "Pattern for python," The Journal of Machine Learning Research, vol. 13, no. 1, pp. 2063–2067, 2012.

[5] B. Sagot, "The lefff, a freely available and large-coverage morphological and syntactic lexicon for french," in 7th international conference on Language Resources and Evaluation (LREC 2010), 2010.

[6] B. New, C. Pallier, L. Ferrand, and R. Matos, "Une base de données lexicales du français contemporain sur internet : Lexique™//a lexical database for contemporary french : Lexique™," L'année psychologique, vol. 101, no. 3, pp. 447–462, 2001.

[7] T. De Smedt and W. Daelemans, "" vreselijk mooi!"(terribly beautiful) : A subjectivity lexicon for dutch adjectives." in LREC, 2012, pp. 3568–3572.

[8] A. Carvalho, A. Levitt, S. Levitt, E. Khaddam, and J. Benamati, "Off-the-shelf artificial intelligence technologies for sentiment and emotion analysis : a tutorial on using ibm natural language processing," Communications of the Association for Information Systems, vol. 44, no. 1, p. 43, 2019.

[9] M. Pontiki, D. Galanis, H. Papageorgiou, I. Androutsopoulos, S. Manandhar, M. Al-Smadi, M. Al-Ayyoub, Y. Zhao, B. Qin, O. De Clercq et al., "Semeval-2016 task 5 : Aspect based sentiment analysis," in International workshop on semantic evaluation, 2016, pp. 19–30.

[10] H. H. Do, P. Prasad, A. Maag, and A. Alsadoon, "Deep learning for aspect-based sentiment analysis : a comparative review," Expert systems with applications, vol. 118, pp. 272–299, 2019.

[11] H. A. Uymaz and S. K. Metin, "Vector based sentiment and emotion analysis from text : A survey," Engineering Applications of Artificial Intelligence, vol. 113, p. 104922, 2022.

[12] Z. Wang, S.-B. Ho, and E. Cambria, "A review of emotion sensing : categorization models and algorithms," Multimedia Tools and Applications, vol. 79, no. 47, pp. 35 553–35 582, 2020.

[13] A. Seyeditabari, N. Tabari, and W. Zadrozny, "Emotion detection in text : a review," arXiv preprint arXiv :1806.00674, 2018.

[14] S. PS and G. Mahalakshmi, "Emotion models : a review," International Journal of Control Theory and Applications, vol. 10, no. 8, pp. 651–657, 2017.

[15] E. Kim and R. Klinger, "A survey on sentiment and emotion analysis for computational literary studies," arXiv preprint arXiv :1808.03137, 2018.

[16] F. A. Acheampong, C. Wenyu, and H. Nunoo-Mensah, "Text-based emotion detection : Advances, challenges, and opportunities," Engineering Reports, vol. 2, no. 7, p. e12189, 2020.

[17] R. A. Calvo and S. Mac Kim, "Emotions in text : dimensional and categorical models," Computational Intelligence, vol. 29, no. 3, pp. 527–543, 2013.

[18] P. Ekman, "Facial expressions in dalgleish t & power m (eds.), handbook of cognition and emotion (pp. 301–320)," 1999.

[19] P. Ekman and D. Cordaro, "What is meant by calling emotions basic," Emotion review, vol. 3, no. 4, pp. 364–370, 2011.

[20] J. A. Russell, "A circumplex model of affect." Journal of personality and social psychology, vol. 39, no. 6, p. 1161, 1980.

[21] ——, "Affective space is bipolar." Journal of personality and social psychology, vol. 37, no. 3, p. 345, 1979.

[22] J. A. Russell, M. Lewicka, and T. Niit, "A cross-cultural study of a circumplex model of affect." Journal of personality and social psychology, vol. 57, no. 5, p. 848, 1989.

[23] J. A. Russell and A. Mehrabian, "Evidence for a three-factor theory of emotions," Journal of research in Personality, vol. 11, no. 3, pp. 273–294, 1977.

[24] A. Mehrabian, "Pleasure-arousal-dominance : A general framework for describing and measuring individual differences in temperament," Current Psychology, vol. 14, no. 4, pp. 261–292, 1996.

[25] I. Bakker, T. Van Der Voordt, P. Vink, and J. De Boon, "Pleasure, arousal, dominance : Mehrabian and russell revisited," Current Psychology, vol. 33, no. 3, pp. 405–421, 2014.

[26] R. Kalitvianski, "Traitements formels et sémantiques des échanges et des documents textuels liés à des activités collaboratives [formal and semantic processing of exchanges and textual documents related to collaborative activities.]," Theses, Université Grenoble Alpes, Mar. 2018, Accessed on Sep. 3, 2021. [Online]. Available : https://tel.archives-ouvertes.fr/tel-01893348

[27] A. T. Latinjak, "The underlying structure of emotions : A tri-dimensional model of core affect and emotion concepts for sports," Revista Iberoamericana de Psicología del Ejercicio y el Deporte, vol. 7, no. 1, pp. 71–88, 2012.

[28] R. Plutchik, "A psychoevolutionary theory of emotions," 1982.

[29] E. Cambria, A. Livingstone, and A. Hussain, "The hourglass of emotions," in Cognitive behavioural systems. Springer, 2012, pp. 144–157.

[30] K. Byron, "Carrying too heavy a load ? the communication and miscommunication of emotion by email," The Academy of Management Review, vol. 33, no. 2, pp. 309–327, 2008, Accessed on Aug. 18, 2021. [Online]. Available : http://www.jstor.org/stable/20159399

[31] J.-E. Kim and K. Johnson, "The Impact of Moral Emotions on Cause-Related Marketing Campaigns : A Cross-Cultural Examination," Journal of Business Ethics, vol. 112, no. 1, pp. 79–90, January 2013, Accessed on Aug. 18, 2021. [Online]. Available : https://ideas.repec.org/a/kap/jbuset/v112y2013i1p79-90.html

[32] M. S.-F. Virginie Rodriguez, "Le contenu des communications relationnelles par email des enseignes : Quelle perception par le consommateur ? [Content of retailers' relational e-mails : what is the consumer's perception ?]," in 20th International Marketing Trends Conference, Venise, Italy, Jan. 2021, Accessed on Aug. 29, 2021.

[33] R. Miller and E. Charles, "A psychological based analysis of marketing email subject lines," in 2016 Sixteenth International Conference on Advances in ICT for Emerging Regions (ICTer), 2016, pp. 58–65.

[34] B. Klimt and Y. Yang, "The enron corpus : A new dataset for email classification research," in Machine Learning : ECML 2004, J.-F. Boulicaut, F. Esposito, F. Giannotti, and D. Pedreschi, Eds. Berlin, Heidelberg : Springer Berlin Heidelberg, 2004, pp. 217–226.

[35] A. Abdaoui, J. Azé, S. Bringay, and P. Poncelet, "FEEL : a French Expanded Emotion Lexicon," Language Resources and Evaluation, vol. 51, no. 3, pp. 833–855, Sep. 2017. [Online]. Available : https://hal-lirmm.ccsd.cnrs.fr/lirmm-01348016

[36] U. Yaqub, S. A. Chun, V. Atluri, and J. Vaidya, "Analysis of political discourse on twitter in the context of the 2016 us presidential elections," Government Information Quarterly, vol. 34, no. 4, pp. 613–626, 2017. [Online]. Available : https://www.sciencedirect.com/science/article/pii/S0740624X17301910

[37] A. Kumar, "An empirical examination of the effects of design elements of email newsletters on consumers' email responses and their purchase," Journal of Retailing and Consumer Services, vol. 58, p. 102349, 2021. [Online]. Available : https://www.sciencedirect.com/science/article/pii/S0969698920313576

[38] A. Mueller and S. Guido, Machine learning avec Python. O'Reilly Media, Inc., 2018.

[39] E. Saravia, H.-C. T. Liu, Y.-H. Huang, J. Wu, and Y.-S. Chen, "CARER : Contextualized affect representations for emotion recognition," in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Brussels, Belgium : Association for Computational Linguistics, Oct.-Nov. 2018, pp. 3687–3697. [Online]. Available : https://www.aclweb.org/anthology/D18-1404

[40] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," arXiv preprint arXiv :1910.10683, 2019.

[41] M. Aiken, "An updated evaluation of google translate accuracy," Studies in Linguistics and Literature, vol. 3, p. p253, 07 2019.

Chapitre 4

Modélisation des newsletters

4.1 Retour sur la modélisation du contenu de newsletters

4.1.1 Motivations pour une nouvelle représentation

Dans le chapitre précédent, nous avons défini la tâche qui consiste à prédire la classe de performance de newsletters. Cependant, si nous avons pu étudier l'impact que peuvent avoir les émotions pour cette tâche, ses caractéristiques seules ne permettent pas de prédire les classes de performance.

Jusqu'à présent, nous avons considéré le texte d'une newsletter comme une seule et même entité. Or, celui-ci n'est pas monolithique : il est structuré en différentes zones et cette structuration participe à l'impression globale suscitée par la newsletter et à son impact sur le lecteur. Par exemple, un bouton rouge au centre de la page, attire davantage l'attention qu'un texte situé en périphérie mentionnant des informations légales. Il semble donc important de considérer, outre le contenu de chacune des zones de texte, ses propriétés graphiques et sa visualisation.

Ainsi, pour aller plus loin dans l'analyse d'une newsletter, il apparaît nécessaire de changer la représentation des données pour modéliser la mise en page, et considérer le contexte graphique dans lequel s'insère le texte. Cette approche consistant à considérer le contexte d'un texte afin d'en préciser l'analyse a déjà été utilisée [1, 2]. Dans ces travaux, les auteurs cherchent à prédire si un message lors d'une conversation en ligne contient du langage abusif. Pour cela ils prennent en compte, non seulement le contenu du message, mais aussi la topologie du graphe associée. Cette double approche a permis d'obtenir des résultats performants sur les tâches de classification de messages haineux. Le graphe est utilisé ici pour modéliser les relations sociales liées à la conversation étudiée. Or, dans notre cas, c'est le contexte graphique qu'il nous faut modéliser sous forme de graphe, et ainsi pouvoir étudier précisément son influence sur la performance des newsletters.

Cette méthode de compréhension d'un document en le découpant en zones est utilisée dans un autre domaine, le DAL (*Document Layout Analysis)* [3]. Ce champ de recherche cherche notamment à détecter les informations au sein d'un document, à reconnaître les tables ou les différents types d'encarts qui le composent tels que titre, image, etc. Certaines approches utilisent des modèles qui considèrent à la fois les informations textuelles (contenu) et celles de

l'image (mise en page) au sein d'architectures d'auto-encodeurs [3, 4, 5, 6]. Une approche un peu plus complexe rajoute également une représentation en graphe associée à des convolutions de graphe [7] exploitées par un auto-encodeur.

Si tous ces modèles sont intéressants dans leur méthodologie, ils se focalisent cependant sur une tâche précise : l'extraction d'informations clés ou KIE (*Key Information Extraction*). Or dans notre cas, le but n'est pas tant d'analyser ou de "comprendre" les informations du document mais plutôt d'en déduire l'impact produit sur le lecteur.

Une autre tâche de DAL consiste à chercher l'ordre d'enchaînement des zones de texte, et donc un potentiel sens de lecture. Or, une étude ayant réalisé un *eye-tracking* sur les mails de *phishing* [8], tend à montrer que, pour ce genre de contenu assez semblable aux newsletters, le regard de l'utilisateur se pose de manière plutôt indifférenciée sur la page et ne suit pas d'ordre de lecture précis.

Ainsi, pour réaliser notre nouvel embedding, nous avons tout comme dans d'autres méthodes [8, 3] utilisé un graphe et des convolutions pour modéliser notre newsletter, et segmenté notre page en différentes zones.

Pour cela, nous avons utilisé un OCR (reconnaissance optique des caractères) pour détecter l'emplacement des zones de texte, puis nous avons proposé un moyen de définir les liens entre ces différentes zones. Notre objectif final est de représenter une newsletter par un graphe complet où chaque nœud représente une zone de texte et auquel nous associons des features. Les liens entre les nœuds doivent modéliser la cohérence globale de la mise en page ; ainsi, par exemple, les zones de texte en périphérie de la newsletter sont représentées par des nœuds en périphérie du graphe.

4.1.2 Mise en oeuvre

Détection des zones de texte

L'une des premières tâches a été de détecter les zones de texte dans la page. Pour cela nous avons utilisé *LayoutLM* l'outil d'OCR [9]. LayoutParser utilise le modèle *Detectron2* [10], qui est conçu pour la détection d'objets dans une scène. Dans notre cas, cela permet de segmenter la page en plusieurs types de zones (cf. Figure 4.1). Puisque nous ne nous intéressons qu'au texte, nous ne conservons ici que les zones labélisées "Text" et "Title".

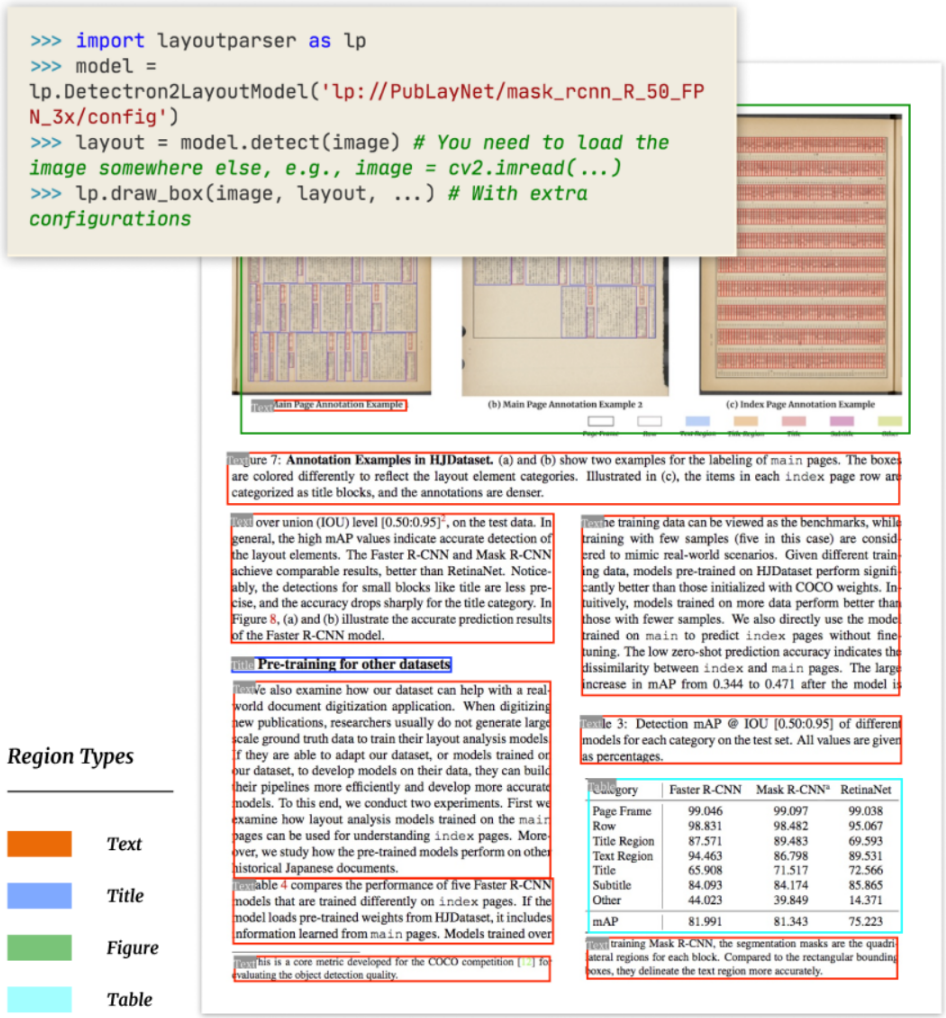


FIGURE 4.1 : Schéma représentant la segmentation proposée par LayoutParser utilisant *Detectron2*. Image tirée du dépôt github de LayoutParser (https ://github.com/Layout-Parser/layout-parser)

Cependant, cette détection peut être défectueuse, il nous faut nous assurer que les zones détectées ne se chevauchent pas, et réadapter la segmentation le cas échéant, en "fusionnant" les "boxes" par la conservation des limites maximum des deux zones se chevauchant, comme illustré en figure 4.2.

Nous avons choisi ce modèle pour pouvoir, si besoin, faire évoluer notre approche pour prendre en compte le caractère multimodal de nos données. En effet, *Detectron2* permet aussi de détecter des zones telles que des images ou des tableaux. Si dans cette thèse nous nous sommes concentrés uniquement sur le texte, cette approche multimodale pourrait faire l'objet de travaux

ultérieurs.

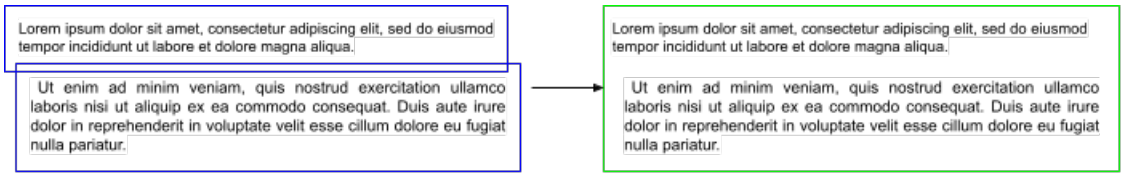
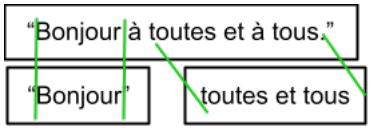


FIGURE 4.2 : Illustration de l'algorithme de correction de la segmentation dans le cas où deux zones détectées se chevauchent

Alignement des zones de texte avec les divisions *HTML*

Par la suite, il nous a fallu associer chacune des zones de texte détectées au texte source correspondant. Pour cela nous avons utilisé le code *HTML* de la page comme source. Dans le fichier *HTML* la page est segmentée en paragraphes à l'aide de divisions ou *div*. Notre tâche d'alignement a consisté à associer à chaque *div*, une zone de texte ou *box*. Chaque *box* est représentée par les coordonnées cartésiennes de ses deux extrémités situées en haut à gauche $M_1(x_1; y_1)$, et en bas à droite $M_2(x_2; y_2)$; l'origine de notre repère étant en haut à gauche. À partir de ces coordonnées, nous avons pu effectuer une OCR ciblée sur la box en utilisant l'outil Tesseract présent dans LayoutParser [9, 11].

Afin de pallier les différences de taille potentielles entre les paragraphes définis par le code *HTML* et ceux reconnus par *Detectron2*, nous avons aussi adapté cette mesure en faisant "coulisser" une fenêtre de la taille de la plus petite chaîne sur la plus grande. Au final, la div associée à une box donnée est celle qui minimise la distance ainsi obtenue. Cet algorithme que nous appellerons "Levenshtein coulissant" est illustré avec un exemple en figure 4.3.



L'algorithme sauvegarde la distance de Levenshtein minimale à mesure que l'on fait “coulisser” la plus petite chaîne de caractère sur l'autre.

Tableau comparatif des algorithmes sur notre exemple

Phrase d'origine : “Bonjour à toutes et à tous.”	partie 1 : “Bonjour”	partie 2 : “toutes et tous”
Distance de Levenshtein	20	13
Distance après application de l'algorithme “Levenshtein_coulissant”	0	4

FIGURE 4.3 : Illustration de l'apport de l'algorithme "Levenshtein coulissant" par rapport à une distance de Levenshtein simple. Notre exemple prend le cas d'une phrase, mais en pratique nous l'appliquons surtout sur des paragraphes.

Cependant, certaines div peuvent ne pas être associées à une box. Cela peut être dû à une non reconnaissance de l'OCR ciblée, ou plus souvent à une différence de segmentation entre la division en paragraphes dans le code *HTML*, et la segmentation graphique déterminée par *Detectron2*. Ainsi il nous a fallu considérer deux types d'objets : les "div" qui sont uniquement définies par le texte dans le code *HTML*, et les "box", auxquelles on a pu attribuer une zone graphique dans l'image de la newsletter.

Nous nous concentrons sur le texte, et donc toute "box" retenue est associée à une "div". C'est pourquoi nous avons construit ces objets selon le diagramme de classes illustré en figure 4.4. L'ensemble de ce processus d'alignement est présenté de manière synthétique en figure 4.5.

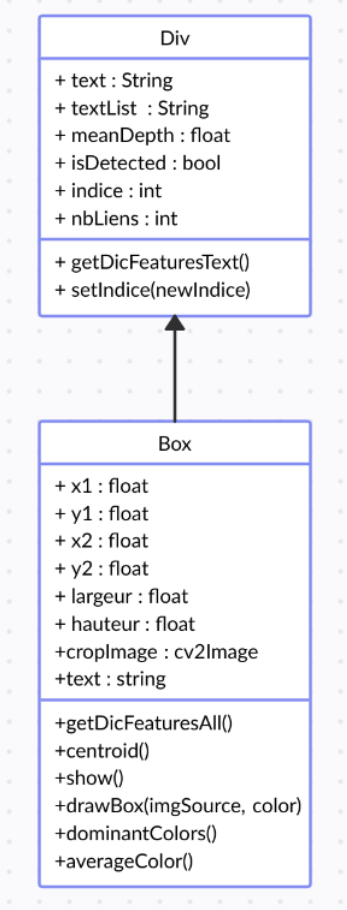


FIGURE 4.4 : Diagramme de classes représentant les objets "box" et "div" référençant respectivement les zones de texte détectées et non détectées

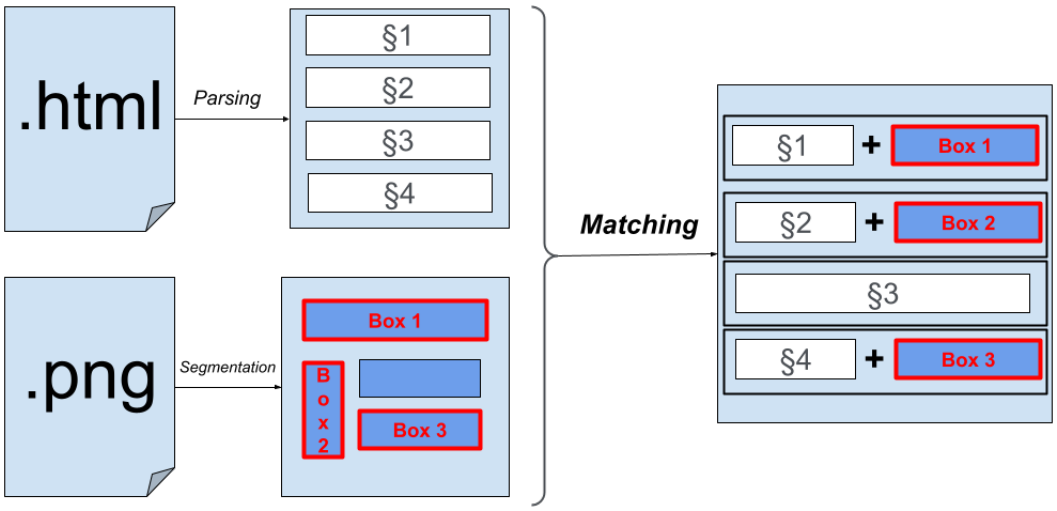


FIGURE 4.5 : Schéma représentant l'alignement entre les textes issus de l'HTML et les zones de texte détectées dans l'image de la newsletter. Dans le cas où on ne peut pas associer de zone graphique au texte, nous ne considérons que le texte.

Construction du graphe

Ces deux types d'objets nous amènent à considérer deux types de noeuds dans notre graphe : ceux correspondant aux zones détectées, et ceux uniquement représentés par le texte issu du code HTML. Ainsi nous avons défini deux conditions de connectivité entre les noeuds : une qui permet de relier entre elles les zones de textes reconnues suivant leur contexte graphique, et une qui permet de relier les paragraphes non détectés au reste du graphe suivant leur emplacement dans le code HTML.

La première de ces définitions permet donc de relier deux zones de texte. Celles-ci sont représentées par des rectangles, et nous avons défini la connectivité comme suit :

Définition 1. *Deux zones de textes sont considérées comme liées si et seulement s'il est possible de tracer un segment entre les centres de leurs rectangles représentatifs sans que ce segment ne passe par une autre zone de texte.*

Afin de faciliter les calculs, et puisque toutes ces zones sont des rectangles, on peut redéfinir cette condition de la manière suivante :

Définition 2. *Deux zones de textes rectangulaires sont considérées comme liées si et seulement s'il est possible de tracer un segment entre leurs centres respectifs sans que ce segment ne coupe une des diagonales des autres zones de texte.*

Une illustration de cette condition est donnée en figure 4.6. Cette condition est assez similaire à la notion de lisibilité introduite par [12] et illustrée en figure 4.7.

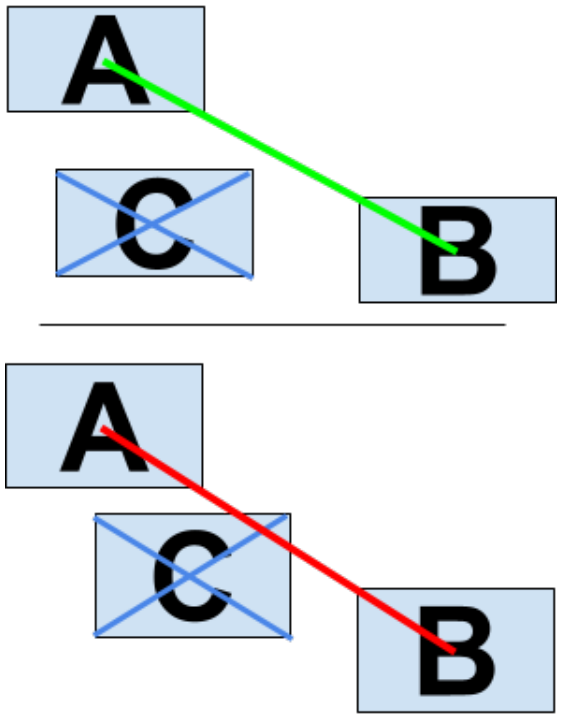


FIGURE 4.6 : Condition de connectivité entre deux zones de texte

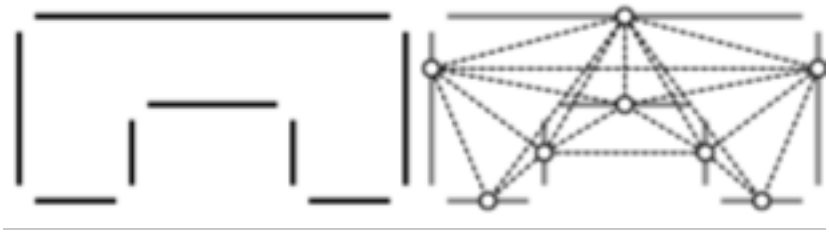


FIGURE 4.7 : Condition de visibilité telle que décrite dans par Locteau et al. [12]

Pour la seconde condition, qui permet de relier les paragraphes non détectés, nous avons défini la connectivité ainsi :

Définition 3. *On considère qu'un paragraphe non détecté est lié aux paragraphes ou zones de texte qui sont liés à la division précédente et suivante dans le code HTML.*

Cette définition de la connectivité, essaie aussi de rendre compte d'une proximité graphique des paragraphes, la proximité des divisions HTML devant a priori être associée à une proximité dans le rendu graphique.

Grâce à ces définitions de la connectivité nous obtenons à la fin un graphe **connexe**, **non orienté** et **hétérogène**. Ces propriétés sont importantes à considérer dans le modèle de classification de graphes que nous verrons ultérieurement. Un exemple de représentation graphique d'une newsletter est donné en figures 4.8 et 4.9. Cependant, avant d'évoquer ce point, il convient de s'attarder sur les différentes features considérées pour la caractérisation des différents noeuds.

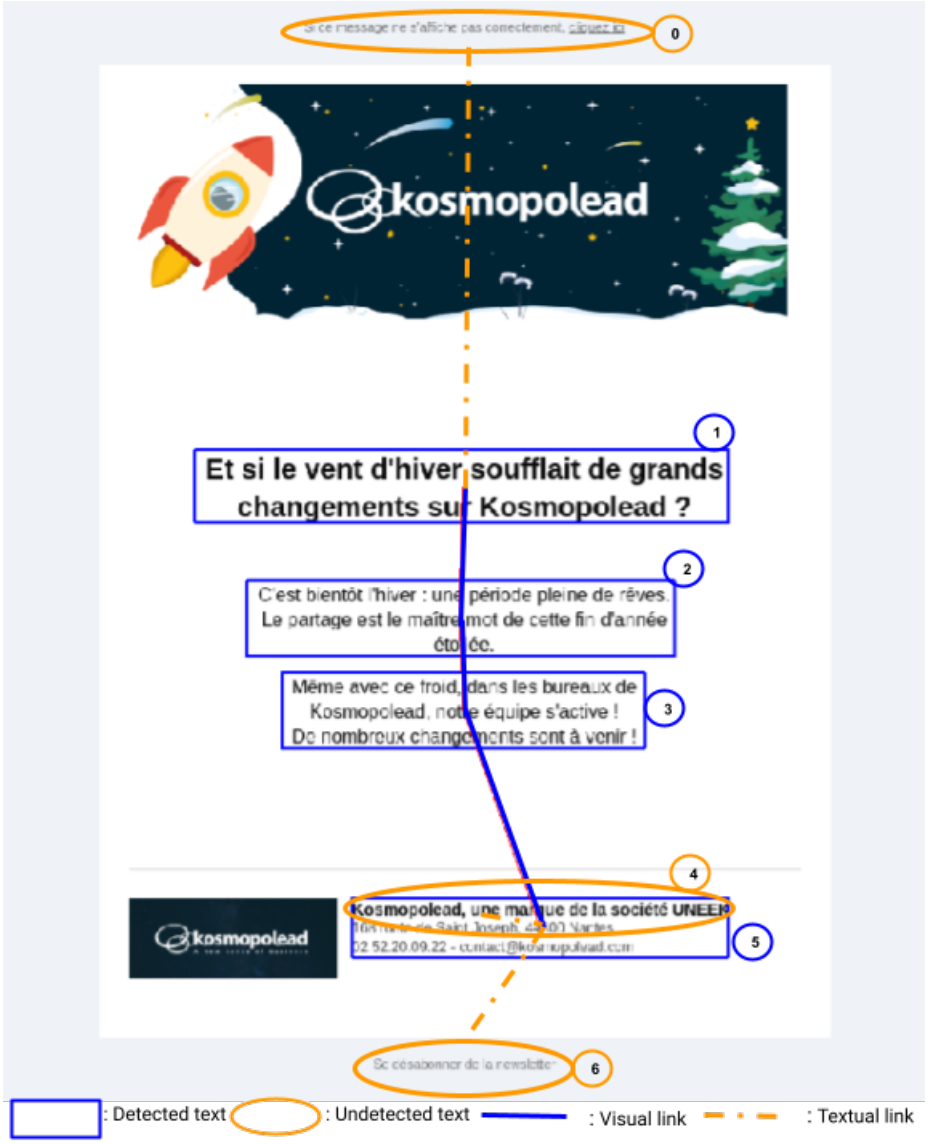


FIGURE 4.8 : Exemple de newsletter, avec en bleu les zones de textes détectées par l'OCR, et en jaune, celles qui ne le sont pas

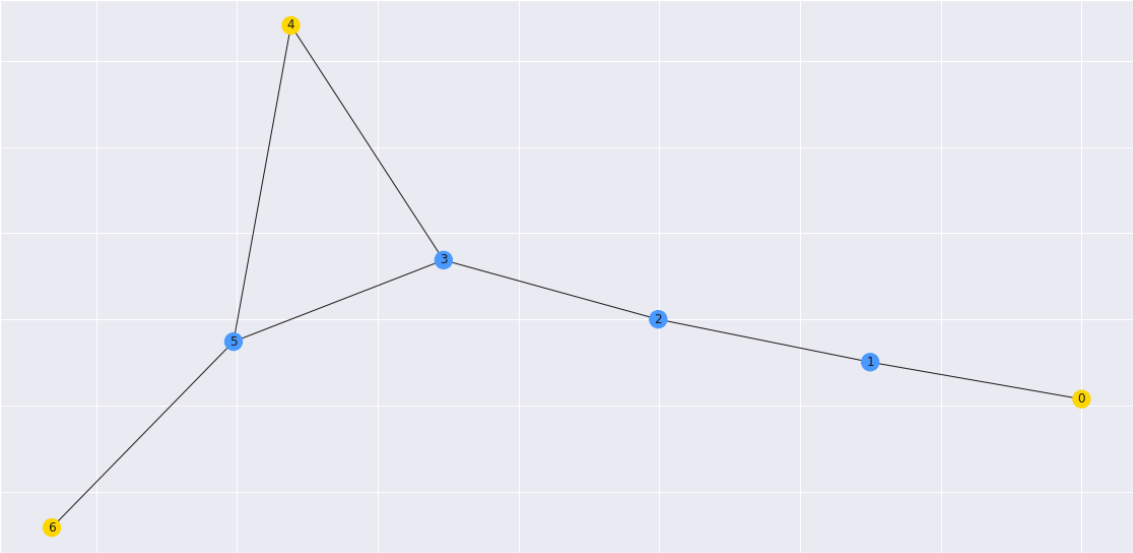


FIGURE 4.9 : Visualisation du graphe modélisant la newsletter donnée en 4.8. Les noeuds bleus représentent les zones détectées, et les noeuds jaunes celles qui ne le sont pas

4.1.3 Ingénierie des caractéristiques (*feature engineering*)

Features textuelles

Ces features sont utilisées pour tous les noeuds, car nous fondons notre méthode sur une analyse du texte. De plus, afin de garantir une forme d'explicabilité si besoin, nous avons privilégié les features pouvant être directement associées à une caractéristique précise du langage, plutôt qu'un embedding abstrait du texte en plusieurs centaines de dimensions. Ainsi nous avons analysé les données textuelles, en prenant en compte les features qui devraient avoir un impact sur la performance des newsletters :

- **Features d'opinion (dim 2) :** Nous appelons ici features d'opinion, les features définies par les scores de polarité et de subjectivité décrits dans le chapitre 3. Celles-ci sont extraites en utilisant l'outil : *Python TextBlob for Natural Language Processing.*

- **Features émotionnelles :** Ces features sont issues des modèles d'extraction d'émotions retenus précédemment (cf. chapitre 3). Ces features sont issues de deux méthodes complémentaires :

  – *FEEL (dim 6) :* 6 features obtenues via le lexique Feel (joie, peur, tristesse, colère, surprise et dégoût).

  - *T5-FR (dim 6) :* 6 features obtenues via l'adaptation du modèle T5 pour détecter les émotions en français (joie, peur, tristesse, colère, surprise et amour)

- **Nombre de liens (dim 1) :** la présence ou non de liens URL est détectée au sein du texte issu du code HTML via des expressions régulières. Le nombre de liens devrait en effet être critique pour étudier la performance d'une newsletter en termes de taux de clics générés. Il est à noter qu'après leur décompte, les liens URL sont retirés du texte pour l'obtention des autres features.

- **Nombre de "mots de spam" (ou *spam words*) (dim 1) :** l'usage de spam words peut grandement influer sur la perception d'un mail. Pour cela nous nous sommes appuyés sur une liste de spam words en français utilisée dans la détection de spams, ou dans l'édition de mails.

- **Longueur du texte (dim 1) :** la longueur du texte est déterminée par son nombre de mots.

- **Profondeur Syntaxique (dim 1) :** afin de modéliser la complexité syntaxique d'une phrase, nous avons pris en compte la profondeur de l'arbre syntaxique de la phrase. Ce dernier est obtenu par l'utilisation de la bibliothèque python *Spacy* [13].

Features graphiques

Ces features sont obtenues par l'analyse de l'image de la zone de texte détectée. Pour garantir la même dimension pour tous les noeuds, nous avons initialisé toutes ces features à 0 dans le cas où la zone de texte n'est pas détectée. Ce type d'initialisation est par ailleurs la méthode privilégiée en cas de données manquantes [14].

- **Coordonnées (dim 4) :** Comme décrit plus haut, il s'agit des coordonnées des points en haut à gauche $(x_1; y_1)$, et en bas à droite $(x_2; y_2)$ de la zone de texte. En effet l'origine du plan cartésien se situant ici dans le coin supérieur gauche de la page, ce sont ces sommets du rectangle que nous avons retenus.

- **Largeur et longueur (dim 2) :** En s'appuyant sur les coordonnées décrites ci-dessus, nous avons calculé la largeur (différence des abscisses) et la longueur (différence des ordonnées) de la zone de texte. Si cela peut apparaître redondant vis-à -vis des coordonnées, ils nous a semblé important d'ajouter ces valeurs, car elles rendent bien compte de l'aire occupée par la zone de texte en question.

- **Couleur moyenne (R, V, B) (dim 3) :** Il s'agit du pixel moyen obtenu en faisant la moyenne de tous les pixels de la zone considérée. Cette valeur donne des informations sur

la visibilité d'une zone de texte, ou encore sur la police qui peut être utilisée. Ainsi un texte écrit en gras aura un pixel moyen plus sombre (si le texte est écrit noir sur blanc) et donc avec des valeurs (R,V,B) plus faibles que le même texte écrit dans une police plus légère.

- **Couleur dominante (R,V,B) (dim 3) :** il s'agit de la couleur dominante de la zone considérée. Pour cela nous avons fait appel à un algorithme de K-moyennes[1]. Cette feature permet de déterminer par exemple si un bouton est d'une couleur assez vive, là où le pixel moyen est bruité par la couleur du texte.

À l'issue de ce travail sur nos données, il nous a fallu encore filtrer à la main les potentiels doublons ou erreurs dans le jeu de données original. Nous avons finalement produit un jeu de 799 newsletters représentées en graphes afin de modéliser les relations graphiques entre les différentes parties de la mise en page. Concrètement cette structure en graphe est codée en utilisant la librairie python *Deep Graph Library* [15] qui propose une formalisation d'un tel type de données, ainsi que des méthodes d'apprentissage adaptées. Ce sont ces méthodes que nous allons détailler dans la partie suivante.

4.2 Convolutions de Graphes

Ces dernières années, les réseaux de neurones de graphes (*Graph Neural Networks*) ou GNN, ont fourni des méthodes permettant de réaliser des avancées dans de nombreux domaines. Si le but ici n'est pas de faire un état des lieux exhaustif des méthodes employées, nous essayerons de dresser un tableau de certaines méthodes animant ce champ de recherche, et les raisons pour lesquelles nous avons choisi une approche particulière. D'autres études proposent des revues plus exhaustives de ce champ de recherche [16, 17, 18, 19].

4.2.1 Présentation du champ de recherche

Les réseaux de neurones de convolution (CNN), sont parmi les premiers modèles à avoir été utilisés en intelligence artificielle, notamment pour l'analyse de document [20]. En effet, elles permettent de prendre en compte des features locales puis de les agréger afin d'en déduire des informations sur la structure globale. Cette capacité à changer d'échelle a permis de nombreuses avancées dans presque tous les domaines auxquels ces méthodes ont pu s'appliquer [21]. Cependant il est aussi apparu que, pour être performants, les CNN ne peuvent s'appliquer que sur des données euclidiennes [16], telles que des images (qui peuvent être vues comme des grilles de pixels en deux dimensions), ou du texte (qui peut être vu comme une suite de mots, ou une

1. Avec K égal à 5, qui représente le nombre de couleurs à détecter. Il est rare de trouver plus de cinq couleurs dans la même portion d'image, et si c'est la cas, nous ne considérons ici que la couleur dominante.

chaîne de caractères, en une dimension). Or, un certain nombre de données sont structurées de manière moins ordonnée (molécules, réseaux sociaux, documents, etc). Ainsi, afin de proposer des solutions aux problématiques attachées à ces données, il est apparu nécessaire de s'intéresser à un moyen d'appliquer ces méthodes de convolution à des données non-euclidiennes [22], et en ce qui nous concerne, les graphes. Ainsi par la suite nous parlerons spécifiquement des réseaux de neurones de graphes ou GNN. En 4.10 nous pouvons voir comment l'idée de la convolution appliquée aux images peut être adaptée aux graphes.

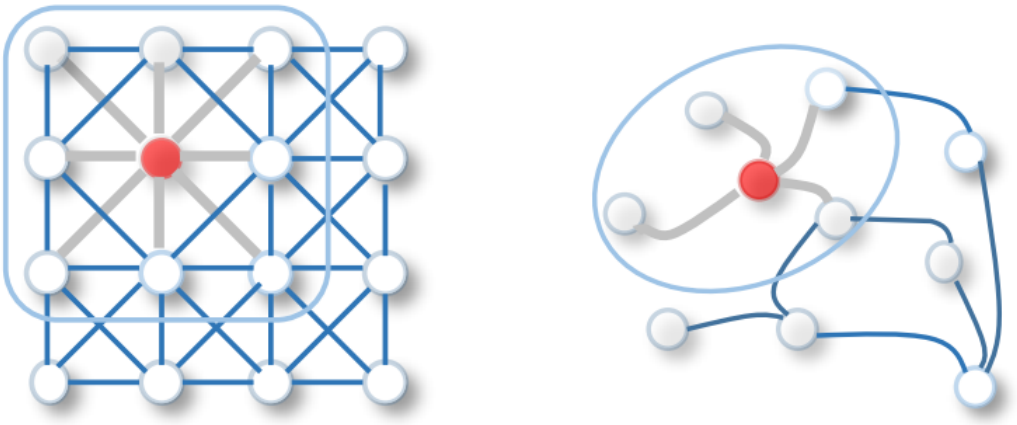


FIGURE 4.10 : A gauche, la représentation d'une convolution sur une image vue comme "graphe grille", à droite le même concept appliqué à un graphe quelconque [17]

4.2.2 Catégorisations des GNN (*Graph Neural Networks*)

Tout d'abord, on peut distinguer les différents types de graphes auxquels on peut appliquer ces méthodes :

- **Les graphes orientés/ non orientés :** l'orientation d'un graphe permet de déterminer le sens de circulation de l'information entre les noeuds d'un graphe. Dans le cas où le graphe est non-orienté, on considère que les liens sont orientés dans les deux sens.

- **Les graphes homogènes/ hétérogènes :** un graphe hétérogène se caractérise par plusieurs types de noeuds (et a fortiori plusieurs types d'arêtes). Pour un nombre n de types de différents noeuds, on peut considérer l'ensemble des liaisons possibles soit 2^n. Il existe cependant certains cas particuliers de graphes où certains types de noeuds ne peuvent être liés qu'à certains types de noeuds (comme les graphes bipartites). A l'inverse un graphe homogène considère que tous les noeuds et liens sont de même type.

- **Les graphes statiques/ dynamiques :** Un graphe dynamique, est un graphe dont les

features des noeuds ou des arêtes sont susceptibles de varier dans le temps. Un graphe qui n'est pas dynamique est considéré comme statique.

Il existe bien sûr d'autres critères issus de la théorie des graphes pour décrire les graphes, mais ceux-ci nous semblent être les principaux à prendre en compte pour choisir la méthode à appliquer. On peut toutefois rajouter comme critère la taille du graphe étudié : certains graphes peuvent être composés de plusieurs millions de noeuds (comme les réseaux sociaux [16] alors que d'autres sont bien plus petits (comme les molécules étudiées en pharmacologie) [23].

Dans notre cas d'étude de newsletters, les graphes que nous étudions sont **petits**, **non-orientés**, **hétérogènes** et **statiques**.

Par la suite, il nous faut aussi prendre en compte la tâche sur laquelle on souhaite faire travailler notre modèle. On peut distinguer trois types de tâches [16, 17] :

- **L'étude des noeuds :** Cette catégorie regroupe toutes les tâches qui cherchent à déduire ou prédire des informations sur les noeuds d'un graphe donné : prédiction de noeuds, plongement des noeuds, classification de noeuds ou encore clustering de noeuds par exemple.

- **L'étude des liens :** Cette catégorie regroupe toutes les tâches qui cherchent à déduire ou prédire des informations sur les arêtes d'un graphe donné : prédiction de liens, prédictions de features sur les liens, etc.

- **L'étude des graphes :** Cette catégorie regroupe les tâches qui visent à déduire des informations sur l'ensemble du graphe, comme la classification de graphes par exemple [18]. Cela peut amener à changer l'architecture globale du modèle, notamment en y ajoutant une fonction de readout qui permet d'agréger les plongements sur les noeuds pour obtenir un plongement du graphe complet.

Dans notre cas, nous cherchons à classifier des newsletters modélisées sous forme de graphes, c'est donc cette troisième catégorie qui nous intéresse. De plus, comme décrit dans le chapitre 3, notre tâche est une classification supervisée dont les labels ont été obtenus suivant les taux de clics associés aux newsletters.

Toutefois il existe une grande variété de méthodes et de modèles que nous ne détaillerons pas ici, mais un arbre regroupant certaines d'entre elles suivant leurs catégories, ainsi que les champs d'applications possibles est présenté en figure 4.11.

Parmi les architectures proposées dans la littérature nous ne décrirons ici que deux familles qui sont parmi les plus employées [16, 17, 19] : les approches spatiales et spectrales.

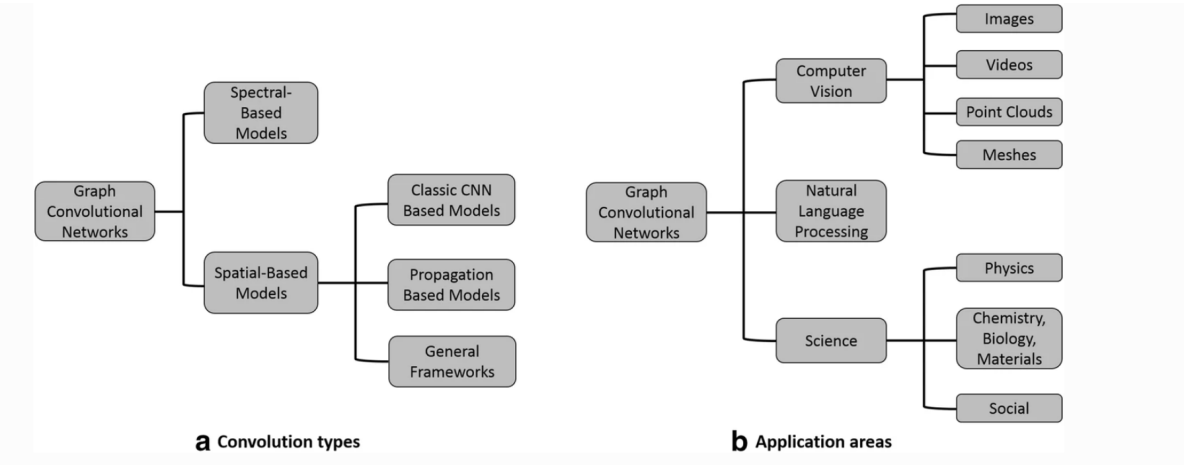


FIGURE 4.11 : Arbre regroupant certaines architectures suivant leurs catégories, ainsi que les champs d'applications possibles [19]

4.2.3 Les approches spatiales

Intuitivement, ce sont celles qui découlent le plus des CNN tels qu' utilisés en traitement d'images. Ces méthodes s'appuient sur la topologie particulière des graphes, en cherchant à propager les informations en suivant les noeuds adjacents. La différence entre ces méthodes porte notamment sur la fonction d'agrégation des informations venant des noeuds voisins. Nous n'expliquerons pas les fondements mathématiques de ces méthodes car cela n'est pas le but ici, et de plus, comme nous allons le voir, certaines méthodes spectrales peuvent par certains aspects être vues comme spatiales [16], cette distinction relevant d'une convention finalement assez vague.

4.2.4 Les approches spectrales

Ces méthodes se fondent sur l'analyse spectrale du graphe en étudiant notamment les valeurs propres de sa matrice d'adjacence. Nous ne rentrerons pas dans les détails mathématiques de ces approches mais nous approfondirons l'une d'elles : les GCN [24].

Considérons un graphe $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ où $|\mathbf{V}| = N$ pour N le nombre de noeuds du graphe, et E l'ensemble de ses arêtes. On notera $\mathbf{A} \in \mathbb{R}^{N \times N}$ sa matrice d'adjacence, et $\mathbf{D} \in \mathbb{R}^{N \times N}$ sa matrice de degrés[2]. Le but ici est de déterminer l'état du graphe avec une entrée de dimension F à l'étape l, représenté par la matrice $\mathbf{H}^{(l)} \in \mathbb{R}^{N \times F}$ à l'issue d'une convolution, pour obtenir la matrice $\mathbf{H}^{(l+1)} \in \mathbb{R}^{N \times F'}$. F' désigne ici la dimension de la sortie[3].

2. Matrice indiquant en ligne i, et colonne j
3. On notera que, dans le cas de la succession de couches cachées de convolution : $F = F'$

La transition se fait via une matrice de poids $\mathbf{W} \in \mathbb{R}^{F \times F'}$, et les valeurs sont mises à jour au fur et mesure de l'apprentissage du modèle.

Considérant les dimensions des matrices énoncées, on peut naïvement définir $H^{(l+1)}$ comme suit :

$$\mathbf{H}^{(l+1)} = \sigma\left(\mathbf{A}\mathbf{H}^{(l)}\mathbf{W}\right)$$

Avec ici σ une fonction d'activation de type *ReLu*, ou *softmax* dans le cas de la couche finale de classification. Cependant, une telle approche entraînerait un très fort sur-apprentissage, et il est nécessaire d'introduire une normalisation qui prenne notamment en compte le degré des noeuds. Il existe plusieurs méthodes de normalisation, mais ici nous ne représenterons que celle donnée par Kipf et al. [24].

$$\mathbf{H}^{(l+1)} = \sigma\left(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W}\right)$$

où les matrices $\tilde{\mathbf{A}}$, et $\tilde{\mathbf{D}}$ sont données par : $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I_N}$ et : $\tilde{\mathbf{D}} = \mathbf{D} + \mathbf{I_N}$, $\mathbf{I_N}$ étant la matrice identité.

L'ajout de la matrice identité revient à considérer que chaque noeud est son propre voisin. De plus, cette méthode ne prenant en compte pour mettre à jour un noeud que les informations venant de son voisinage direct, la méthode GCN est clairement liée à la topologie du graphe, et donc, même si celle-ci est classée comme une méthode spectrale, elle est parfois aussi considérée comme méthode spatiale [19]. L'approche par GCN partage d'ailleurs de nombreux points communs avec GraphSAGE [25], une méthode spatiale qui peut être vue comme une version inductive de GCN, sous la condition que sa fonction d'agrégation est la moyenne [16]. Cependant l'entraînement peut parfois être très coûteux en termes de mémoire pour de grands graphes, ce qui n'est pas notre cas ici.

Enfin, il nous faut considérer les propriétés spécifiques de nos graphes exprimées plus haut. Celle-ci nous poussent à considérer une variante du modèle GCN, le modèle R-GCN [26].

4.2.5 Le modèle R-GCN

Le modèle GCN décrit précédemment peut aussi être décrit par l'équation suivante :

$$h_i^{l+1} = \sigma\left(\sum_{j \in N_i} \frac{1}{c_i} W^{(l)} h_j^{(l)}\right) \tag{4.1}$$

avec pour chaque noeud i un ensemble de voisins N_i, et c_i la constante de normalisation associée.

Le modèle R-GCN, pour *Relational Graph Convolutionnal network*, permet de prendre en compte différents types de noeuds et de liens. Ainsi l'équation (4.1) devient :

$$h_i^{l+1} = \sigma\left(W_0^{(l)} h_i^{(l)} + \sum_{r \in R} \sum_{j \in N_i^r} \frac{1}{c_{i,r}} W_r^{(l)} h_j^{(l)}\right) \tag{4.2}$$

On peut voir que si dans l'équation (4.1), les matrices de poids sont les mêmes, dans l'équation (4.2), chaque type de liens est associé à sa propre matrice de lien. Ainsi, la propagation de l'information qui met à jour le plongement des noeuds de la couche l à la couche $(l+1)$, est pondérée différemment selon le type de liens qui relie les noeuds.

Une fois toutes les convolutions effectuées, comme nous l'avons vu plus tôt, il nous faut déduire un plongement du graphe (notre tâche étant une classification de graphes) à partir du plongement des noeuds. Cela se fait via la fonction de *readout*, qui ici, est une moyenne des plongements des noeuds :

$$h_G = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} h_v \tag{4.3}$$

avec h_G la représentation du graphe $\mathbf{G}$, et V l'ensemble de ses noeuds. Une fois le plongement du graphe obtenu, nous le passons à travers une couche MLP[4], pour réaliser la classification binaire voulue. Afin de tester l'impact des convolutions, nous avons testé diverses architectures en faisant varier le nombre de couches de convolution. L'architecture complète de notre modèle est illustrée en figure 4.12.

4. Multi Layer Perceptron

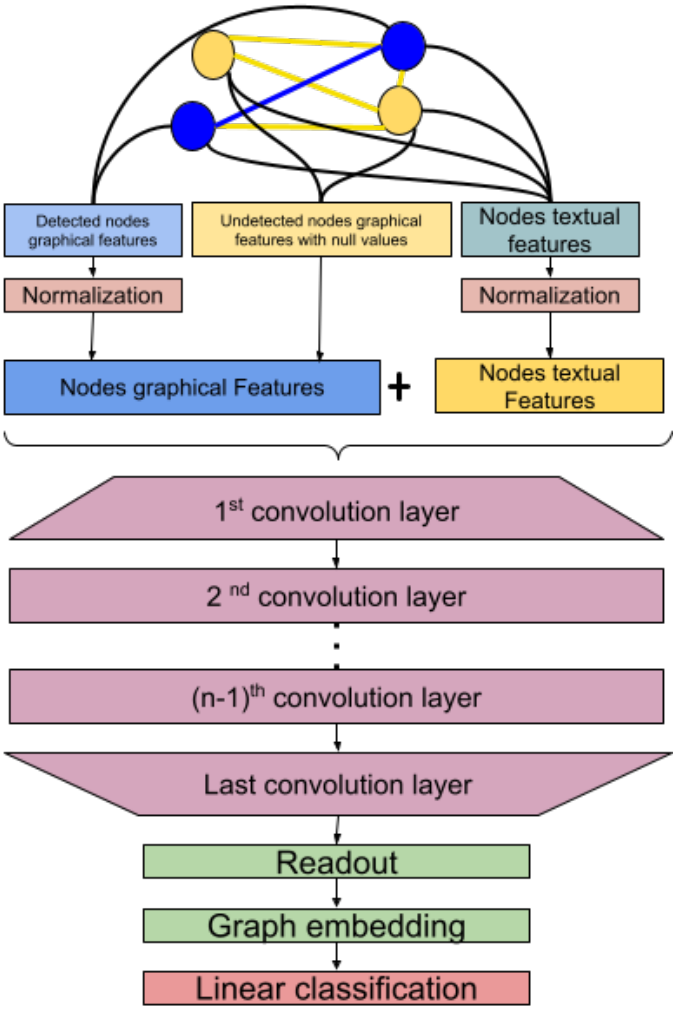


FIGURE 4.12 : Architecture de notre modèle visant à classifier les newsletters représentées en graphes

4.3 Résultats

Dans cette partie nous présentons nos résultats d'expériences, où nous avons testé plusieurs modèles en faisant varier certains paramètres. Ainsi, nous avons tout d'abord fait varier le nombre de couches de convolution de l'architecture pour étudier l'impact du nombre de couches. Par la suite, nous avons ajouté un mécanisme d'attention pour tenter d'améliorer l'architecture, nous avons de même fait varier le taux dropout de cette architecture pour optimiser notre modèle. Enfin, nous avons réalisé une étude par ablation sur les différents types de données pour

savoir ce qui influe le plus entre les features textuelles et visuelles. [5]

4.3.1 Influence de la Convolution

Pour évaluer si notre représentation graphique apporte effectivement une plus-value à l'analyse des newsletters, nous avons opté pour un test de différentes architectures, en faisant varier le nombre de couches de convolution. Ce nombre peut aller de 0 à 6 couches, 0 correspondant à notre baseline, l'information de chaque noeud ne parcourant pas le graphe, et donc ne tenant pas compte de sa structure globale. Nous nous sommes de plus arrêtés à 6 maximum car cela devrait être suffisant compte tenu de la taille relativement petite des graphes considérés ; ainsi avec 6 couches de convolution, toutes les informations de tous les noeuds ont pu atteindre chacun des noeuds. Au total, cela fait 7 architectures à tester. Cependant, il est important de noter qu'ajouter des couches de convolution augmente grandement le temps d'apprentissage des modèles.

Pour réaliser nos apprentissages de modèles, nous avons subdivisé notre jeu de données en 3 parties : 70% des données sont attribuées au jeu d'apprentissage, 15% au jeu de validation et 15% au jeu de test. Nous avons réalisé cette subdivision de manière aléatoire simple, et nous avons réalisé ceci de manière distincte plusieurs fois afin de compenser le nombre plutôt restreint de nos données. Ici, chacune des sept architectures est entraînée sur douze différentes subdivisions[6]. En figure 4.13, nous pouvons voir les différentes courbes d'apprentissage avec les courbes moyennes, pour chaque architecture, sur 300 époques.

5. Les résultats présentés ici sont difficilement comparables avec ceux présentés au chapitre 3, du fait de retravaille des données entre temps.

6. Ces 12 subdivisions sont les mêmes pour les 7 architectures afin de rendre cohérentes les comparaisons des modèles.

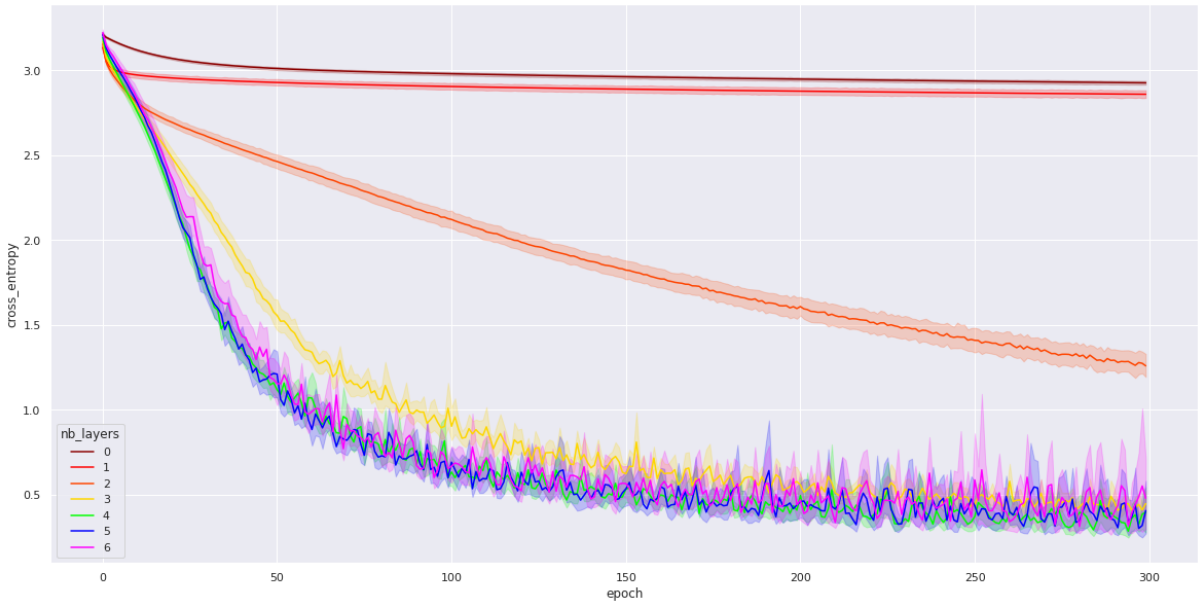


FIGURE 4.13 : Courbes d'apprentissage moyen pour chaque nombre de couches de convolutions.

D'après nos expériences, plus on ajoute de couches de convolution, plus le modèle arrive à apprendre vite, et converge donc vers des valeurs plus intéressantes. Cependant, cela semble aussi augmenter le risque de sur-apprentissage. Ainsi, on peut observer dans les dernières époques, une forte instabilité dans l'apprentissage. Une mesure de l'efficacité de ces modèles par nombre de couches de convolution est donnée en table 4.1. Cette table prend en compte plusieurs métriques afin de compenser le biais qui consiste à considérer la catégorie *"est une bonne newsletter"* comme étant la classe positive de notre tâche de classification binaire, comme vu dans le chapitre 3.

Nombre de couches	*Accuracy*	*Précision*	*Rappel*	*F1-Score*	*ROC-AUC*
0	0.553	0.544	0.549	0.545	0.674
1	0.548	0.540	0.546	0.540	0.671
2	0.614	0.618	0.582	0.596	0.705
3	**0.665**	**0.658**	0.678	0.662	0.709
4	0.658	0.643	**0.701**	**0.663**	0.702
5	0.657	0.645	0.680	0.658	0.720
6	0.657	0.645	0.674	0.655	**0.727**

TABLE 4.1 : Mesure selon plusieurs métriques de la performance de prédiction de la classe de performance en faisant varier le nombre de couches de convolution, avec 12 jeux d'apprentissage/validation/test différents. Le modèle avec 0 couche de convolution est considéré comme une baseline.

Nous constatons alors, que la classification est améliorée avec le nombre de couches de convolution. Cependant, si le nombre de couches de convolution cachées est trop élevé, l'apprentissage est moins stable, comme montré en figure 4.13. Cela peut être lié à la stagnation des résultats, les meilleures performances n'étant pas atteintes avec le nombre maximal de couches de convolution (6), mais plutôt avec 3 ou 4 couches. Ainsi il semble qu'ajouter plus de couches de convolution n'apporte pas beaucoup. Ce problème dit de l'*over-smoothing* est décrit par Oono et Suzuki [27].

Cependant, on peut aussi mesurer l'instabilité de l'apprentissage, en observant l'écart-type de chacune de nos métriques d'évaluation. En effet les résultats précédents sont des moyennes et cela nous aidera par la suite à déterminer si les différences de performance entre architectures sont effectivement significatives ou non. Les mesures de l'écart-type sont visibles dans le tableau 4.2.

Nombre de couches	*std Accuracy*	*std Précision*	*std Rappel*	*std F1-Score*	*std ROC-AUC*
0	0.047	0.059	0.049	0.046	0.047
1	0.048	0.082	0.067	0.063	0.043
2	0.061	0.086	0.057	0.055	0.053
3	0.051	0.074	0.082	0.051	0.058
4	0.053	0.072	0.119	0.070	0.051
5	0.036	0.056	0.067	0.037	0.044
6	0.032	0.053	0.078	0.043	0.037

TABLE 4.2 : Écart type de chacune des métriques d'évaluation présentée en table 4.1

On peut voir que l'écart-type est d'environ 0.05 pour chaque jeu d'apprentissage. Puisqu'il est coûteux en temps d'entraîner des modèles avec un grand nombre de couche, ces résultats nous incitent à considérer les architectures avec 3 couches de convolution ou plus comme étant relativement équivalentes. Ainsi, du fait de l'augmentation importante du temps d'apprentissage à mesure que le nombre de couches de convolution augmente, c'est l'architecture à 3 couches que nous avons conservée pour étudier les autres paramètres.

4.3.2 Influence de l'Attention

La construction d'un embedding de graphe à partir de l'embedding de chacun des noeuds est une étape cruciale, réalisée par la fonction *readout*. Dans les expériences précédentes, nous avons simplement utilisé une fonction de moyenne sur les noeuds. Nous testons ici les gains apportés par une autre fonction de *readout*, telle que l'*Attentive FP readout*.

Cette méthode inspirée par les travaux sur les réseaux d'attention adaptés aux graphes [22, 28], a été initialement conçue pour analyser les molécules afin d'en prédire les propriétés, en représentant les molécules comme des graphes dont les sommets sont les atomes. Or, nos données partagent plusieurs points communs avec les molécules : les graphes sont plutôt petits,

avec plusieurs types de noeuds, et nous sommes intéressés autant par l'aspect global que par les dépendances particulières entre les noeuds. Dans certains cas, ajouter une couche d'attention pour agréger l'information dans chaque noeud semble prometteur [29].

Nous avons utilisé l'implémentation de l'attention proposée dans la librairie python *dgllife* [30]. Cette méthode fut originellement employée dans le cadre de recherche de propriétés de molécules [31]. Pour cela on introduit un "super-noeud" virtuel, connecté à chacun des noeuds du graphe, par des liens pondérés par de l'attention. Le calcul réalisé par la fonction de readout est défini comme suit :

$$h_v^k = GRU^{k-1}(C_v^{k-1}, h_v^{k-1}) \tag{4.4}$$

avec GRU^{k-1} (Gated Recurrent Unit) la fonction de mise à jour à l'itération $k-1$, qui prend en entrée le contexte d'attention, C_v^{k-1} du noeud cible v et son état précédent h_v^{k-1}. La figure 4.14 compare les courbes d'apprentissages de deux versions de notre modèle : une classique avec 6 couches de convolution, et l'autre avec le même nombre de couches mais une fonction de readout utilisant ce mécanisme d'attention.

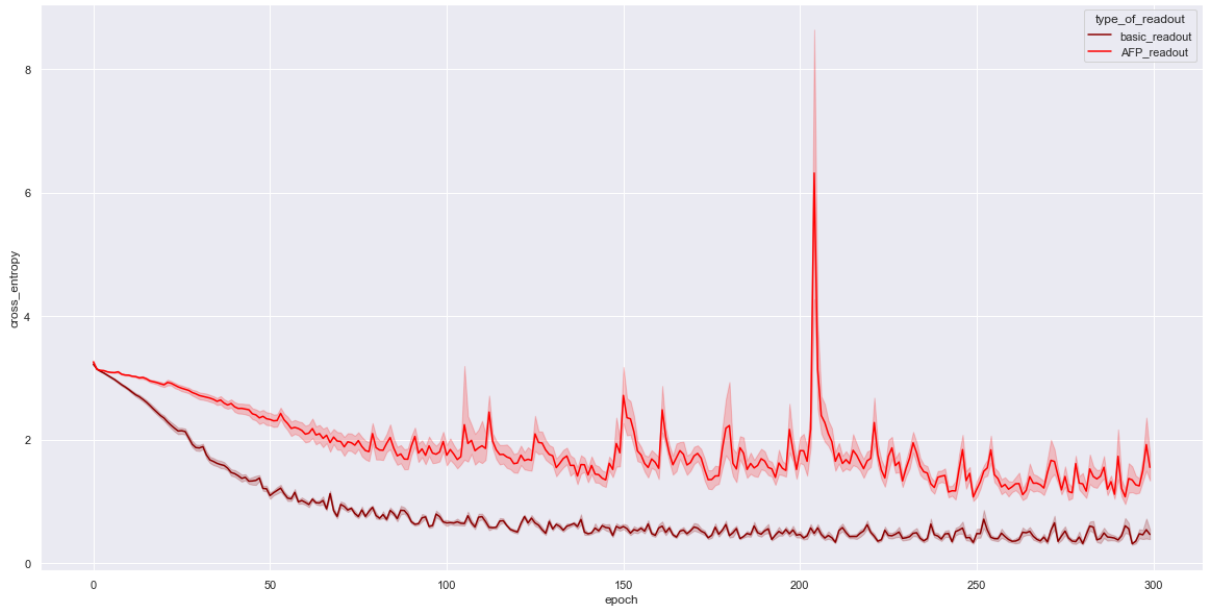


FIGURE 4.14 : Courbes d'apprentissage moyennes pour 6 couches de R-GCN mais avec différentes fonctions readout.

Il apparaît que l'ajout de l'attention est dommageable pour notre tâche, car il diminue grandement les capacités d'apprentissage du modèle, notamment en aggravant le problème de

sur-apprentissage. Ainsi le F1-score peut passer de 0.7 à 0.2. Ces résultats sont à mettre en relation avec les travaux qui montrent que l'ajout de l'attention est d'un apport négligeable, ou peut même diminuer les performances dans certaines circonstances. Ajouter cette couche d'attention a augmenté ici l'instabilité du modèle. Ces problèmes peuvent éventuellement être résolus par l'optimisation de certains hyperparamètres [32, 33] ou bien en améliorant l'architecture classique de *graphical attention network* ou GAT [34].

Ces résultats nous invitent à considérer une architecture plus stable de R-GCN, avec 3 couches. Celle-ci, comme on l'a vu plus tôt, produit d'assez bons résultats et permet aussi de réduire le temps d'apprentissage. Les courbes d'apprentissage de ces modèles sont données en figure 4.15.

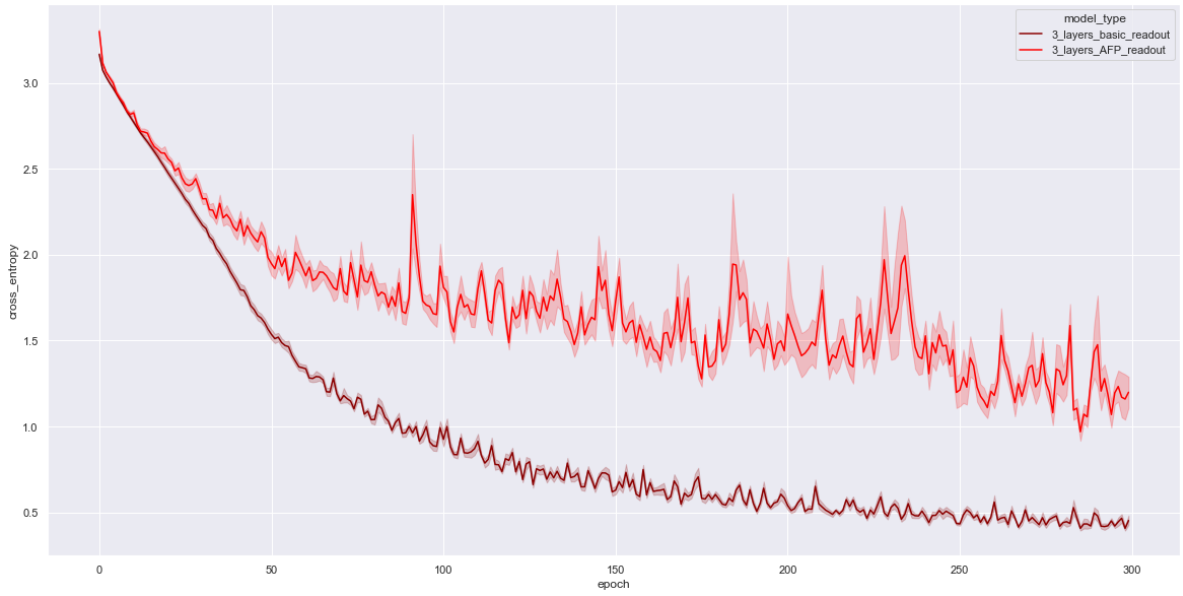


FIGURE 4.15 : Courbes d'apprentissage moyennes pour 3 couches de convolution, avec différentes fonctions de readout.

Même si le processus d'apprentissage est légèrement plus stable, le readout par attention dégrade toujours les résultats. Comme mentionné plus haut, cette méthode est assez sensible aux hyperparamètres, il est donc raisonnable de supposer qu'une optimisation de ces hyperparamètres améliorerait ces résultats.

Avantages du readout d'attention sur l'interprétabilité des prédictions

Si les résultats sont certes bien moins bons avec cette étape d'attention, celle-ci reste néanmoins utile pour aider à expliquer la décision du modèle, et à ce titre elle ne devrait pas être négligée. Puisque notre but est de permettre une aide à l'édition de newsletter, être capable d'indiquer à

l'éditeur du mail quelles parties du mail influent le plus sur sa performance prédite est intéressant dans le cadre d'une interaction homme-machine, voire autant que la performance du modèle considéré.

La figure 4.16 reprend l'exemple donné dans les figures 4.8 et 4.9 précédentes, avec les noeuds colorés en fonction des poids d'attention. Cela permet de visualiser les parties sur lesquelles le modèle s'est focalisé pour établir sa prédiction.

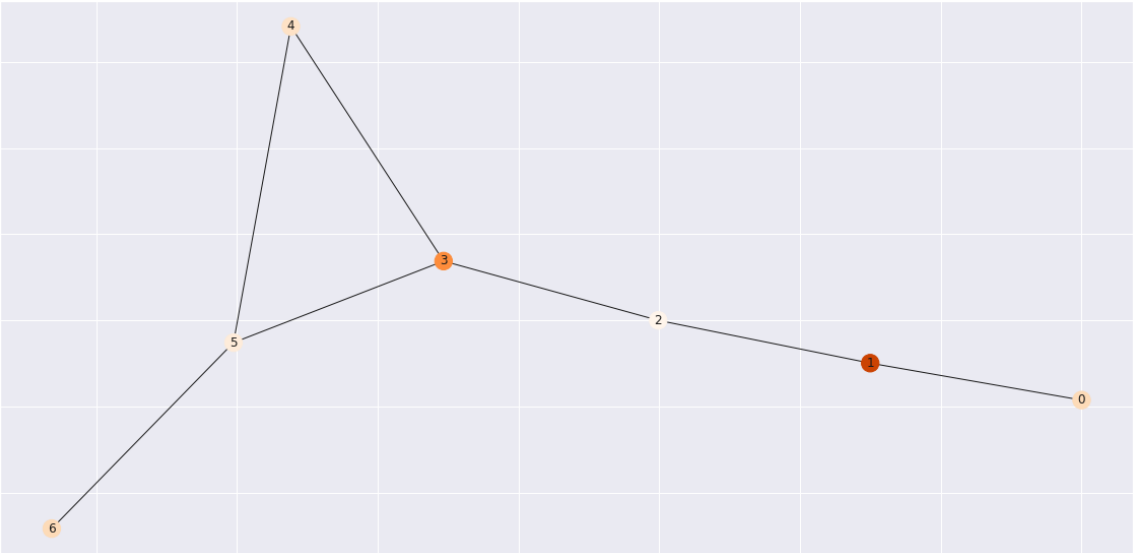


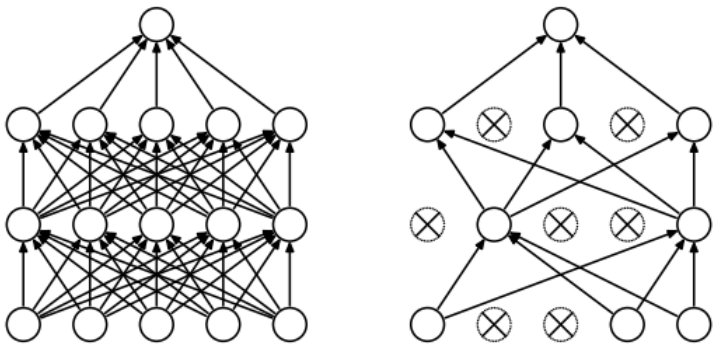
FIGURE 4.16 : Graphe représentant le même exemple de newsletter que les figures 4.8 et 4.9, mais les noeuds sont ici colorés suivant les poids d'attention assignés à chaque noeud lors du readout.

Il apparaît que le noeud 1, représentant le titre de la newsletter, est celui qui capte le plus l'attention du modèle. À l'inverse, les noeuds en périphérie sont plutôt ignorés et reçoivent moins d'attention. S'il faut rester prudent quant à l'interprétation de ces résultats, et sans vouloir donner des intentions anthropomorphiques au modèle, il est intéressant en termes de modélisation, de remarquer que ces placements de l'attention sont assez cohérents comparativement à l'attention humaine sur un document. Enfin, mettre en valeur dans l'outil d'édition les régions recevant le plus d'attention permettrait d'aider l'utilisateur à améliorer les newsletters en se focalisant sur certains points du document.

4.3.3 Influence du dropout

Parmi les hyperparamètres de la fonction de readout utilisant de l'attention, nous avons cherché à affiner le réglage du dropout, afin d'améliorer nos performances. Cette méthode qui consiste à

ignorer aléatoirement certaines connexions entre des noeuds d'un réseau de neurones, est assez utilisée pour résoudre des problèmes de sur-apprentissage [35]. Une illustration de ce mécanisme est donnée en figure 4.17.



(a) Standard Neural Net (b) After applying dropout.

FIGURE 4.17 : Illustration du mécanisme de dropout [35]

Par défaut ce paramètre est à 0 afin de prendre en compte toute l'information disponible. Cependant cela peut conduire à un sur-apprentissage par la redondance de certaines informations, et pour cela, nous avons testé des modèles utilisant le readout d'attention et pour lesquels seul le dropout change. Les courbes d'apprentissage de ces modèles sont en figure 4.18.

FIGURE 4.18 : Courbes d'apprentissage moyennes pour 3 couches de convolution utilisant le readout d'attention, mais avec un taux de dropout variable. Les moyennes sont réalisées selon 5 jeux d'apprentissages, de validation et de test différents. Ici les courbes s'arrêtent à 150 époques, le sur-apprentissage rendant le graphe illisible au-delà. La courbe grise représente la courbe d'apprentissage du modèle n'utilisant pas le readout d'attention.

On remarque que, quel que soit le taux de dropout employé, les courbes d'apprentissage ne diffèrent pas ; ce qui tend à montrer que ce paramètre n'a que peu ou pas d'influence sur l'apprentissage du modèle. De plus, il apparaît que le modèle par défaut (avec un taux de dropout à 0) soit l'option la plus stable. Si on pouvait s'attendre à une forte instabilité pour des taux de dropout élevés (80% de connexions ignorées étant évidemment beaucoup trop), il est assez décevant que les taux de dropout plus raisonnables n'apportent pas beaucoup. Ces considérations sont aussi à mettre en perspective avec la courbe de la même architecture, mais sans le readout d'attention.

Ainsi, nous pouvons en conclure que changer l'hyperparamètre de dropout ne permet pas de compenser les pertes de performances d'apprentissage dues à l'usage d'un readout avec attention. Les résultats de ces différents modèles comparés à la même architecture sans dropout, sont visibles dans le tableau 4.3.

Ces résultats montrent que quelle que soit la métrique d'évaluation considérée, sauf le rappel, ajouter un taux de dropout ne fait que diminuer les performances du modèle. Pour ce qui est du rappel, la bonne performance peut s'expliquer par le biais introduit en considérant la classe "bonne newsletter" comme classe "positive" dans la classification binaire. Le modèle a alors

TABLE 4.3 : Performances des modèles avec un readout attention, suivant différent taux de dropout. Ces résultats sont les moyennes de métriques calculées sur 5 modèles de même architecture entraînés selon différentes séparations du jeu de données 4.18, 4.15, et 4.14.

Taux de dropout	*Accuracy*	*Précision*	*Rappel*	*F1-score*	*ROC AUC*
0.2	0.640	0.669	0.548	0.595	0.683
0.3	0.622	0.646	**0.601**	0.620	0.688
0.4	0.620	0.655	0.569	0.606	0.681
0.5	0.607	0.664	0.502	0.564	0.684
0.6	0.624	0.643	0.600	0.620	0.680
0.7	0.632	0.662	0.571	0.612	0.678
0.8	0.605	0.638	0.518	0.571	0.631
0 (baseline)	**0.647**	**0.674**	0.586	**0.623**	**0.736**

tendance à classer trop de newsletters comme "bonnes" et voit son taux de faux négatifs (i.e de newsletters classées comme "mauvaises" par erreur) diminuer, et donc le taux de rappel diminue. Ce biais n'apparaît pas dans les autres métriques.

De manière générale, on peut conclure que l'ajout d'un dropout pour atténuer l'impact négatif du readout avec attention n'est pas concluant.

4.3.4 Influence des features

Par rapport aux travaux présentés dans le chapitre 3, nous avons pris en compte ici beaucoup de nouvelles features. Il paraît donc pertinent d'en étudier les impacts individuels sur la classification de newsletters. Si auparavant les features représentant l'ensemble de la newsletter pouvaient être obtenues en amont de l'apprentissage du modèle, ce n'est plus le cas ici. En effet, avec notre nouvelle architecture, les features en amont s'appliquent à certaines parties de la newsletter mais son plongement global n'est obtenu qu'à l'issue de l'application de ou des couches de convolution puis de leur agrégation via la fonction readout. Cependant, il nous est possible de déduire l'impact des features *post hoc* en réalisant une étude par ablation. Cela consiste à comparer les résultats du ou des modèles en enlevant une ou plusieurs features. Puisque nous avons deux catégories principales de features, nous pouvons déjà comparer les résultats obtenus jusqu'à présent avec les résultats de modèles entraînés uniquement en tenant compte des features graphiques ou textuelles. Les performances de ces modèles sont présentées dans le tableau 4.4.

On observe que, si pour certaines métriques d'évaluation, c'est la combinaison des deux types de features qui donne les meilleurs résultats, pour d'autres, les meilleurs modèles sont ceux qui ne tiennent compte que des features graphiques. Cela pourrait s'interpréter comme la prédominance de ces features sur les features textuelles pour cette tâche de classification. Cela tend à renforcer notre hypothèse selon laquelle prendre en compte le caractère graphique d'un texte est autant, voire plus important que la prise en compte de son contenu lorsque l'on cherche

TABLE 4.4 : Résultats de modèles ayant la même architecture (trois couches de convolution, readout simple), entraînés selon de jeux de features différents. Ces résultats sont les moyennes de métriques réalisée sur 5 modèles selon différentes séparations du jeu de données. Ces séparations sont les mêmes que pour les figures 4.18, 4.15, 4.14, et le tableau 4.3

Features considérées	*Accuracy*	*Précision*	*Rappel*	*F1-score*	*ROC AUC*
Textuelles seulement	0.622	0.609	**0.758**	0.669	0.669
Graphiques seulement	0.655	0.683	0.612	0.644	0.713
Toutes	**0.692**	**0.716**	0.668	**0.690**	**0.732**

à maximiser le taux de clics d'une newsletter.

4.4 Conclusions et perspectives

Les newsletters sont des documents qui sont organisés selon des zones de texte, et des images. A ce titre, ignorer le contexte visuel du texte qui les compose semble être une erreur. Dans notre cas nous ne nous intéresserons qu'aux zones de texte, l'ajout d'images relevant d'autres champs de recherches sur l'apprentissage profond, et rajouter une modalité ne ferait que complexifier une approche exploratoire déjà assez complexe. Toutefois cela pourrait être une évolution future de notre approche. Ainsi ici, nous avons décrit notre proposition de modélisation de newsletter afin de prendre en compte ce contexte graphique des zones de texte.

Pour cela, nous avons segmenté les newsletters selon leur mise en page en utilisant de la reconnaissance de caractère, et en associant lorsque possible les zones de textes visuelles avec les paragraphes issus du fichier *HTML* de l'email. Afin de rendre compte des influences et relations de ces zones de textes entre elles, nous avons choisi d'utiliser une représentation par graphes, chaque noeud représentant une des zones de textes reconnue ou non. Les noeuds sont reliés suivant une condition de lisibilité qui relie les zones proches dans la mise en page. Chacune des zones de texte peut alors être analysée séparément comme un texte, à l'aide de features textuelles, et lorsque cela est possible à l'aide de features graphiques. Après une application de plusieurs couches de convolution, nous effectuons notre tâche de classification. Il en résulte, tout d'abord, que par rapport à un modèle ne comprenant aucune couche, et donc qui ne prend pas en compte les influences des zones de textes les unes par rapport aux autres, les modèles avec 3 couches ou plus obtiennent de meilleurs résultats.

Cependant il nous faut aussi considérer l'explicabilité du modèle, afin de permettre une interaction homme-machine d'aide à l'édition de newsletters. Pour cela nous avons considéré l'application d'une couche d'attention, à la suite des convolutions, pour passer des plongements de noeuds à celui du graphe. En effet, il devient alors possible de déterminer quels noeuds du graphe, et donc quelles zones de la newsletters influent le plus sur la prédiction. En revanche,

si cet ajout offre des perspectives d'explicabilité, il détériore de beaucoup les performances de la classification. Pensant que cela pouvait être dû au bruit que cette couche rajoute au modèle, nous avons tenté de régler le problème au travers du taux de dropout appliqué à la couche de classification. Or cela n'a mené à rien de concluant, et conserver ce taux à 0 paraît être la meilleure option. Toujours dans le but de renforcer l'explicabilité de notre approche, nous avons réalisé une étude par ablation, en comparant les résultats obtenus lorsque l'on ne prend respectivement que les features textuelles des graphes d'un côté, et uniquement les features graphiques d'un autres. Il en résulte que les modèles entraînés en prenant les features textuelles et graphiques seules, sont moins bons que lorsque que lorsque celles-ci sont combinées. De plus, une analyse plus fine des résultats nous indique que les features textuelles ont tendance à biaiser la classification vers la classe des "bonnes" newsletters.

Ainsi, nous avons proposé une nouvelle approche d'analyse de document, en prenant en compte à la fois le contenu, et l'aspect visuel des différentes zones de texte qui le composent. Cependant cela ne permet pas pour le moment une mise en application où un taux de bonnes prédictions de 80% minimum serait attendu. C'est pourquoi des travaux supplémentaires permettraient de compléter cette approche exploratoire : entraînement de modèles supplémentaires, étude en ablation plus fine, test d'autres paramètres, test d'autres architectures...

Pour cela, nous envisageons de rendre public le jeu de graphes ainsi que le code source afin de permettre à d'autres de reproduire ces résultats et de les améliorer. Toutefois, l'utilisation de graphe semble prometteuse. En effet, elle permet de contextualiser le texte, mais aussi de manière plus générale de prendre en compte des informations hétérogènes, et ajouter un nouveau type de noeuds dans notre modèle pouvant prendre en compte les images serait possible. L'usage de graphe ouvre en effet de multiples perspectives pour le traitement automatique du langage, en permettant la combinaison d'informations hétérogènes [36].

BIBLIOGRAPHIE

[1] E. Papegnies, V. Labatut, R. Dufour, and G. Linares, "Graph-based features for automatic online abuse detection," in International conference on statistical language and speech processing. Springer, 2017, pp. 70–81.

[2] N. Cecillon, V. Labatut, R. Dufour, and G. Linarès, "Abusive language detection in online conversations by combining content-and graph-based features," Frontiers in big Data, vol. 2, p. 8, 2019.

[3] Ł. Garncarek, R. Powalski, T. Stanisławek, B. Topolski, P. Halama, M. Turski, and F. Graliński, "Lambert : layout-aware language modeling for information extraction," in International Conference on Document Analysis and Recognition. Springer, 2021, pp. 532–547.

[4] A. R. Katti, C. Reisswig, C. Guder, S. Brarda, S. Bickel, J. Höhne, and J. B. Faddoul, "Chargrid : Towards understanding 2d documents," arXiv preprint arXiv :1809.08799, 2018.

[5] T. I. Denk and C. Reisswig, "Bertgrid : Contextualized embedding for 2d document representation and understanding," arXiv preprint arXiv :1909.04948, 2019.

[6] P. Zhang, Y. Xu, Z. Cheng, S. Pu, J. Lu, L. Qiao, Y. Niu, and F. Wu, "Trie : end-to-end text reading and information extraction for document understanding," in Proceedings of the 28th ACM International Conference on Multimedia, 2020, pp. 1413–1422.

[7] W. Yu, N. Lu, X. Qi, P. Gong, and R. Xiao, "Pick : processing key information extraction from documents using improved graph learning-convolutional networks," in 2020 25th International Conference on Pattern Recognition (ICPR). IEEE, 2021, pp. 4363–4370.

[8] J. McAlaney and P. J. Hills, "Understanding phishing email processing and perceived trustworthiness through eye tracking," Frontiers in Psychology, vol. 11, p. 1756, 2020.

[9] Z. Shen, R. Zhang, M. Dell, B. C. G. Lee, J. Carlson, and W. Li, "Layoutparser : A unified toolkit for deep learning based document image analysis," arXiv preprint arXiv :2103.15348, 2021.

[10] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019.

[11] R. Smith, "An overview of the tesseract ocr engine," in Ninth international conference on document analysis and recognition (ICDAR 2007), vol. 2. IEEE, 2007, pp. 629–633.

[12] H. Locteau, S. Adam, E. Trupin, J. Labiche, and P. Héroux, "Symbol spotting using full visibility graph representation," in Workshop on Graphics Recognition, 2007, pp. 49–50.

[13] M. Honnibal and I. Montani, "spaCy 2 : Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing," 2017, to appear.

[14] N. Ipsen, P.-A. Mattei, and J. Frellsen, "How to deal with missing data in supervised deep learning ?" in ICML Workshop on the Art of Learning with Missing Values (Artemiss), 2020.

[15] M. Wang, L. Yu, D. Zheng, Q. Gan, Y. Gai, Y. Zihao, M. Li, J. Zhou, Q. Huang, C. Ma, Z. Huang, Q. Guo, H. Zhang, H. Lin, J. Zhao, J. Li, A. Smola, and Z. Zhang, "Deep graph library : Towards efficient and scalable deep learning on graphs," 09 2019.

[16] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks : A review of methods and applications," AI Open, vol. 1, pp. 57–81, 2020.

[17] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," IEEE transactions on neural networks and learning systems, vol. 32, no. 1, pp. 4–24, 2020.

[18] F. Errica, M. Podda, D. Bacciu, and A. Micheli, "A fair comparison of graph neural networks for graph classification," arXiv preprint arXiv :1912.09893, 2019.

[19] S. Zhang, H. Tong, J. Xu, and R. Maciejewski, "Graph convolutional networks : a comprehensive review," Computational Social Networks, vol. 6, no. 1, pp. 1–23, 2019.

[20] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

[21] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," nature, vol. 521, no. 7553, pp. 436–444, 2015.

[22] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning : going beyond euclidean data," IEEE Signal Processing Magazine, vol. 34, no. 4, pp. 18–42, 2017.

[23] O. Wieder, S. Kohlbacher, M. Kuenemann, A. Garon, P. Ducrot, T. Seidel, and T. Langer, "A compact review of molecular property prediction with graph neural networks," Drug Discovery Today : Technologies, vol. 37, pp. 1–12, 2020.

[24] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," arXiv preprint arXiv :1609.02907, 2016.

[25] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," Advances in neural information processing systems, vol. 30, 2017.

[26] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. v. d. Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in European semantic web conference. Springer, 2018, pp. 593–607.

[27] K. Oono and T. Suzuki, "Graph neural networks exponentially lose expressive power for node classification," arXiv preprint arXiv :1905.10947, 2019.

[28] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," arXiv preprint arXiv :1710.10903, 2017.

[29] M. F. Demirel, S. Liu, S. Garg, and Y. Liang, "An analysis of attentive walk-aggregating graph neural networks," 2021. [Online]. Available : https://arxiv.org/abs/2110.02667

[30] M. Li, J. Zhou, J. Hu, W. Fan, Y. Zhang, Y. Gu, and G. Karypis, "Dgl-lifesci : An open-source toolkit for deep learning on graphs in life science," ACS Omega, 2021.

[31] Z. Xiong, D. Wang, X. Liu, F. Zhong, X. Wan, X. Li, Z. Li, X. Luo, K. Chen, H. Jiang et al., "Pushing the boundaries of molecular representation for drug discovery with the graph attention mechanism," Journal of medicinal chemistry, vol. 63, no. 16, pp. 8749–8760, 2019.

[32] B. Knyazev, G. Taylor, and M. Amer, "Understanding attention in graph neural networks," in Proceedings of the ICLR RLGM Workshop, 1905.

[33] M. Klabunde and F. Lemmerich, "On the prediction instability of graph neural networks," arXiv preprint arXiv :2205.10070, 2022.

[34] S. Brody, U. Alon, and E. Yahav, "How attentive are graph attention networks ?" 2021. [Online]. Available : https://arxiv.org/abs/2105.14491

[35] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout : a simple way to prevent neural networks from overfitting," The journal of machine learning research, vol. 15, no. 1, pp. 1929–1958, 2014.

[36] V. Labatut, "Combining heterogeneous information."

CONCLUSION

Cette thèse qui porte sur "*l'Analyse d'opinion pour l'optimisation de la relation client*" explore plusieurs problématiques. Tout d'abord dans le premier chapitre, nous avons essayé de développer une interaction directe entre le client et une base de données SQL, via une requête en langage naturel.

Ces travaux se sont heurtés à plusieurs problèmes qui ont mené à des impasses, en particulier la difficulté de passer de l'anglais au français et le manque de données. Cela nous a incités à nous tourner vers une autre forme de communication et de relation client du point de vue d'une entreprise ou structure amenée à utiliser les outils de UNEEK : les campagnes d'email ou newsletters. Dans le chapitre 2, nous avons présenté pourquoi elles sont un canal de communication intéressant pour étudier la relation client, et comment nous avons abordé le problème de leur analyse. Nous nous sommes focalisés sur les newsletters envoyées via l'outil proposé par *UNEEK*. Ces données sont spécifiques et importantes pour la gestion de la relation client, leur utilisation intensive justifie que l'on s'intéresse à la tâche qui consiste à optimiser leur efficacité et donc leur rédaction. Il restait à trouver une approche pour en optimiser l'envoi et c'est ce que nous avons vu dans les chapitres suivants.

Notre tâche se propose d'aider à l'édition de newsletters en prévoyant la réaction du receveur du mail, avant leur envoi. Dans le chapitre 3, nous nous sommes notamment intéressés à l'analyse d'opinion et d'émotions, et nous avons présenté un état de l'art des différents modèles d'analyse d'émotions et d'opinion dans le langage. Nous avons ensuite redéfini notre tâche, pour l'assimiler à un problème de classification des newsletters en "bonnes" ou "mauvaises", suivant le taux de clics généré par la campagne. Ces données (newsletters et statistiques associées) sont enregistrées sur les serveurs *UNEEK*. En testant la résolution de cette tâche via des approches d'analyse des émotions nous avons pu déterminer que les émotions suscitées par le texte des newsletters ont un impact au mieux neutre et au pire négatif ; ce qui est en accord avec la littérature du domaine du marketing.

Dans le chapitre 4, nous avons proposé une modélisation des newsletters qui prend en compte leur mise en page. Nous avons fait le choix d'une représentation sous forme de graphes et, ainsi, notre tâche a évolué en une tâche de classification de graphes.

Si les modèles testés ne permettent pas pour le moment une mise en production (les résultats montrent au mieux un taux de bonne prédiction de 66%), ils semblent néanmoins montrer que notre approche est prometteuse : elle prend en compte la disposition des textes de la newsletter et leur mise en forme graphique (couleur du texte, du fond, taille des fenêtres, etc), et elle

apporte une aide à la prédiction de performance de newsletters.

En raison de la grande spécificité de la tâche (données en français, contexte propre à l'entreprise, etc), ces travaux sont en grande partie exploratoires. Il demeure de nombreuses pistes d'amélioration qui n'ont pu être testées dans cette thèse : en premier lieu l'augmentation de la quantité de données, qui limite à l'heure actuelle les approches par apprentissage de modèles, ainsi que la validité des interprétations des résultats de ces modèles. Au delà des données d'apprentissage, des pistes d'amélioration concernent également l'approche par graphes pour laquelle de nombreuses architectures de modèles venant d'autres champs de recherche restent à tester. De plus, notre approche permet d'ajouter des noeuds propres à l'analyse d'images et donc d'offrir une analyse multimodale des newsletters.

De plus cette méthodologie originale pourrait s'appliquer à d'autres champs de recherche nécessitant de mettre en relation des données avec différentes modalités, pour concevoir un embedding global, comme ici où nous avons mis en relation des features textuelles et graphiques au sein d'un même embedding.

Enfin, pour faciliter les tests et reproductions des résultats, nous prévoyons de mettre le code ainsi que le jeu de graphes utilisés à disposition de la communauté scientifique.

COLLEGE MATHSTIC
DOCTORAL BRETAGNE
BRETAGNE OCEANE

Université Bretagne Sud

Titre : Analyse d'opinion pour l'optimisation de la relation client

Mot clés : Intelligence Artificielle, Traitement automatique du langage, Convolution de graphes, Apprentissage profond

Résumé : Cette thèse a pour sujet l'*"Analyse d'opinion pour l'optimisation de la relation client"*; elle a été réalisée dans le cadre d'une convention *CIFRE*, en partenariat avec la société *UNEEK*, qui propose des services de gestion de relation client.

Le sujet est vaste et recoupe de nombreux domaines, comme le requêtage de base de données en langage naturel, l'apprentissage profond, le traitement automatique du langage, l'analyse et la détection d'émotions ou le marketing et les sciences sociales. Nos travaux abordent essentiellement deux problématiques : l'interrogation en français de données tabulaires et l'étude de newsletters.

Dans une première partie, nous explorons la possibilité de concevoir une interface de requêtes permettant d'interroger en langage naturel les bases de données tabulaires de l'entreprise.

Ensuite, nous expliquons comment et pourquoi nous avons recentré nos travaux sur l'étude d'un canal de communication particulier de l'entreprise vers ses contacts : les newsletters.

Nous étudions d'abord comment les émotions transmises par le texte des newsletters, influent sur leur perception, et comment les résultats de cette étude peuvent aider à leur rédaction et leur édition.

Ensuite, nous proposons une modélisation des newsletters sous forme de graphes hétérogènes, permettant de prendre en compte les aspects visuels des zones de textes et leur disposition dans la newsletter, en plus de leur contenu. Nous utilisons des techniques d'apprentissage profond telles que les réseaux de convolution de graphe, et les techniques d'attention pour prédire la performance des newsletters.

Cette modélisation originale des newsletters a produit des résultats encourageants pour la tâche de prédiction considérée. L'approche pourrait être approfondie dans de futurs travaux pour prendre en compte d'autres composantes significatives des newsletters, en particulier les images. De plus cette modélisation pourrait être appliquée pour d'autres études multi-modales.

Title: Opinion analysis for customer relationship optimization

Keywords: Artificial Intelligence, Natural language processing, Graph Convolutional Networks, Deep Learning

Abstract:

The subject of this thesis is "Opinion analysis for customer relationship optimization"; it was carried out within the framework of a CIFRE agreement, in partnership with the company UNEEK, which offers customer relationship management services.

The subject is vast and cuts across many fields, such as databases interrogation in natural language, deep learning, automatic language processing, analysis and detection of emotions or marketing and social sciences. Our work essentially addresses two issues: the querying of tabular data in French and the study of newsletters.

In a first part, we explore the possibility of designing a query interface allowing to query in natural language the tabular databases of the company.

Then, we explain how and why we have refocused our work on the study of a particular communication channel from the company to the contacts: newsletters.

We first study how the emotions transmitted by the text of newsletters influence their perception, and how the results of this study can help in their writing and editing.

Then, we propose a modeling of newsletters in the form of heterogeneous graphs, allowing to take into account the visual aspects of text areas and their layout in the newsletter, in addition to their content. We use deep learning techniques such as graph convolution networks, and attention techniques to predict newsletter performance.

This original modeling of newsletters produced encouraging results for the prediction task considered. The approach could be further developed in future work to take into account other significant components of newsletters, in particular images. Furthermore, this modelisation could be applied in other multi-modal studies