Détection des discours haineux dans les réseaux sociaux : apport des expressions polylexicales

Nicolas Zampieri

UNIVERSITÉ DE LORRAINE

Inria

Loria
Laboratoire lorrain de recherche en informatique et ses applications

Thèse de Doctorat

Nicolas Zampieri

Mémoire présenté en vue de l'obtention du
grade de Docteur de l'Université de Lorraine
Mention Informatique

École doctorale: IAEM

Unité de recherche : Laboratoire Lorraine de Recherche en Informatique et ses Applications
UMR 7503

Soutenue le 13 décembre 2023
Thèse N°:

Détection des discours haineux dans les réseaux sociaux : apport des expressions polylexicales

Composition Du Jury

Directrice de thèse :	**Irina ILLINA**, HDR, Maître de conférence, Université de Lorraine, LORIA-INRIA, France
Rapporteur :	**Richard DUFOUR**, Professeur, Université de Nantes, LS2N, France
Rapporteuse :	**Farah BENAMARA**, HDR, Maître de conférence, Université Paul Sabatier, IRIT, France
Présidente :	**Agatha SAVARY**, Professeur, Université Paris-Saclay, LISN, France
Examinateur :	**Frédéric BECHET**, Professeur, Université d'Aix-Marseille, LIS, France
Examinatrice :	**Claire GARDENT**, Directeur de Recherche, CNRS, LORIA-INRIA, France
Invité :	**Dominique FOHR**, Chargé de recherche (Retraité), CNRS, LORIA-INRIA, France
Invité :	**Carlos Ramisch**, Maître de conférence, Université d'Aix-Marseille, LIS, France
Invité :	**Christophe CERISARA**, Chargé de recherche, CNRS, LORIA-INRIA, France

Remerciements

Je tiens à exprimer ma profonde gratitude envers toutes les personnes qui ont contribué de près ou de loin à la réalisation de cette thèse. Ce travail de recherche représente le résultat de nombreuses années d'efforts et n'aurait pas été possible sans le soutien, l'encouragement et la collaboration de plusieurs individus et institutions.

Je tiens surtout à remercier ma famille et mes amis qui m'ont soutenu avec amour et compréhension. Votre encouragement constant et votre confiance en moi ont été mes plus grandes sources de motivation.

Table des matières

Chapitre 5
Identification des expressions polylexicales dans les tweets

Chapitre 6
Les expressions polylexicales pour la détection de la parole haineuse

Table des figures

Liste des tableaux

Glossaire

BERT : *Bidirectional Encoder Representations from Transformers*
BERT-FT : BERT *fine-tuned*
Bi-LSTM : *Bidirectionnal Long Short-Term Memory*
BIO : *Begin-Inside-Outside*
Bi-RNN : *Bidirectionnal Recurrent Neural Network*
CBOW : *Continuous Bag of Words*
CM : Casse des mots
CNN : *Convolutional Neural Network*
CRF : *Conditional Random Fields*
DiMSUM : *Detecting Minimal Semantic Units and their Meanings*
DPH : Détection de la parole haineuse
DPH-2B : Système de Détection de la Parole Haineuse à deux branches neuronales
DPH-3B : Système de Détection de la Parole Haineuse à trois branches neuronales
DSM : DiMSUM
DualCL : *Dual Contrastive Learning*
EmoVec : Vecteur d'émojis
EP : Expression polylexicale
GRU : *Gated Recurrent Units*
LLM : *Large Language Model*
LR : *Logistique Regression*
LSR : *Lexical Semantic Recognition*
LSTM : *Long Short-Term Memory*
MH : Mot haineux
MLM : *Masked Language Model*
Moy : Moyenne des scores de similarité cosinus
NSP : *Next Sentence Prediction*
$\mathbf{P_{aléa}}$: Paire de tweets avec une méthode aléatoire
PARSEME : *PARSing and Multi-word Expressions*
POS : *Part-of-Speech*
$\mathbf{P_{sim}}$: Paire de tweets avec une méthode basée la similarité cosinus
PSM : PARSEME
Punct : Ponctuation
RNN : *Recurrent Neural Network*
RoBERTa : *Robustly Optimized BERT Aproach*
sBERT-ACH : *Sentence BERT* avec Apprentissage par Contraste pour la Haine
ST : Streusle
STS : *Semantic Textual Similarity*
SVM : *Support Vector Machine*

TAL : Traitement automatique des langues
TF-IDF : *Term Frequency - Inverse Document Frequency*
UL : Unité lexicale
USE : *Universal Sentence Encoder*
VM : Vote majoritaire

Résumé

Au cours de ces dernières décennies, l'utilisation d'Internet s'est répandue de manière exponentielle, notamment avec l'avènement des réseaux sociaux. Cependant, cette augmentation de l'utilisation des médias sociaux a engendré une prolifération des messages néfastes, tels que les discours haineux. Les discours haineux sont des formes de communication dégradantes qui ciblent spécifiquement un individu ou un groupe, pouvant mener à des menaces et à des actes de violence. Les modèles basés sur l'apprentissage profond sont rapidement devenus une solution pour détecter les discours haineux. Cependant, ces modèles nécessitent une quantité considérable de données d'apprentissage pour atteindre des niveaux de performance élevés. De plus, certaines caractéristiques spécifiques peuvent être essentielles pour détecter les discours haineux. Dans cette thèse, nous explorons deux principaux axes de recherches visant à améliorer les performances de classification des discours haineux. Nos approches sont expérimentées sur quatre ensembles de données distincts, permettant ainsi une évaluation approfondie de leur efficacité dans la détection des discours haineux.

Dans la première partie de notre étude, nous proposons d'améliorer la détection des discours haineux en intégrant des caractéristiques spécifiques dans un réseau de neurones basé sur les plongements (*embeddings* en anglais) de phrase. Plus précisément, nous examinons l'incorporation de caractéristiques telles que la casse des mots, les émojis, les mots présents dans un dictionnaire de termes haineux, les parties du discours et la ponctuation. Notre approche consiste à développer un réseau neuronal qui intègre ces caractéristiques au niveau des mots, en complément des plongements de phrase. Nous montrons que l'utilisation des émojis améliorent significativement les performances de détection des discours haineux.

Ensuite, nous nous intéressons à l'intégration de caractéristiques peu explorées jusqu'à présent dans la détection des discours haineux : les expressions polylexicales. Nous menons une étude approfondie sur la robustesse des systèmes d'identification de ces expressions dans les tweets. Cette étude nous permet d'évaluer différents systèmes pour l'identification des expressions polylexicales dans les tweets, afin d'annoter automatiquement les jeux de données destinés à la détection des discours haineux en termes d'expressions polylexicales. Nous montrons que les systèmes basés sur l'apprentissage profond surpassent ceux basés sur les dictionnaires dans cette tâche. De plus, nous proposons un système en deux étapes qui combine à la fois le système basé sur l'apprentissage profond et celui basé sur les dictionnaires. Celui-ci surpasse les deux systèmes existants pour l'identification des expressions polylexicales dans les tweets. Puis, nous développons deux réseaux de neurones qui s'appuient sur les plongements de phrase et intègrent ces informations sur les expressions polylexicales de manière différente. Nous montrons une amélioration significative des performances sur la tâche de détection des discours haineux en utilisant les informations sur les expressions polylexicales.

Dans la seconde partie, nous explorons différentes approches d'apprentissage pour améliorer les performances sur la détection des discours haineux. Nous nous intéressons tout d'abord à l'impact de l'apprentissage multi-tâche. Pour cela, nous proposons un réseau neuronal basé sur l'attention multi-têtes pour l'apprentissage multi-tâche. Notre système est conçu pour apprendre simultanément deux tâches : la détection des discours haineux et l'identification des expressions polylexicales. Nous montrons qu'apprendre en simultané l'attention sur les expressions polylexicales et sur les discours haineux améliorent la détection de ces derniers.

Ensuite, nous explorons l'utilisation de l'apprentissage par contraste pour la détection des discours haineux. Notre approche consiste à appliquer cet apprentissage de manière supervisée. L'objectif principal est d'apprendre des plongements de phrase tels que les tweets appartenant à la même classe soient rapprochés (selon la distance cosinus), tandis que les tweets appartenant à des classes différentes soient éloignés. Pour atteindre cet objectif, nous proposons différentes méthodes pour créer des paires de tweets d'apprentissage. Nous proposons également différentes méthodes de décision basée sur la distance cosinus pour prédire les classes. Nous montrons que nos approches permettent d'obtenir des performances équivalentes à celles de l'apprentissage classique pour la classification des discours haineux. Cependant, l'aspect intéressant de notre approche réside dans le fait que les plongements de phrase générés à partir de l'apprentissage par contraste permettent d'observer une meilleure séparation des classes dans un espace vectoriel observable, par rapport aux plongements générés après un apprentissage classique pour la classification des discours haineux.

Abstract

Over the last few decades, internet use has increased dramatically, particularly with the emergence of social networks. However, this explosion in social media activity has led to the spread of harmful messages, including hate speech. Hate speech is a type of degrading communication aimed specifically at an individual or group, which can take the form of threats and acts of violence. Deep learning models have quickly become a means of detecting hate speech. However, these models require a substantial amount of training data to achieve high levels of performance. In addition, some specific features may be essential for the hate speech detection. In this thesis, we examine two main areas of research aimed at improving hate speech detection performance. We test our methods on four separate datasets, allowing us to thoroughly evaluate their effectiveness in detecting hate speech.

In the first part of our study, we propose to enhance the hate speech detection by incorporating specific features into a neural network based on sentence embeddings. More specifically, we examine the incorporation of features such as word case, emojis, words present in a hate speech lexicon, part-od-speech, and punctuation. Our approach involves developing a neural network that integrates these features at the word level, in addition to sentence embeddings. We demonstrate that the use of emojis significantly improves the performance of the hate speech detection.

Next, we focus on the integration of underexplored features in hate speech detection : multiword expressions. We conduct an in-depth study on the robustness of systems for identifying these expressions in tweets. This study allows us to assess different systems for identifying multiword expressions in tweets, in order to automatically annotate datasets intended for hate speech detection in terms of multiword expressions. We demonstrate that deep learning-based systems outperform dictionary-based ones in this task. Furthermore, we propose a two-step system that combines both the deep learning-based and dictionary-based systems. This system outperforms the two existing systems for identifying multiword expressions in tweets. Then, we develop two neural networks that rely on sentence embeddings and integrate these multiword expressions information in different ways. We show a significant improvement in performance on the hate speech detection task using multiword expression information.

In the second part, we explore different learning approaches to enhance performance in hate speech detection. First, we investigate the impact of multi-task learning. For this purpose, we propose a multi-head attention-based neural network for multi-task learning. Our system is designed to simultaneously learn two tasks : hate speech detection and multiword expression identification. We demonstrate that jointly learning attention mechanisms for multiword expressions and hate speech enhances the detection of the latter.

Next, we explore the use of contrastive learning for hate speech detection. Our approach involves applying this learning method in a supervised manner. The main objective is to learn sentence embeddings such that tweets belonging to the same class are brought closer together (according to cosine distance), while tweets from different classes are pushed apart. To achieve this goal, we propose different methods to create pairs of training tweets. We also suggest various cosine distance-based decision methods to predict classes. We demonstrate that our approaches achieve performance equivalent to traditional learning for hate speech classification. However, the intriguing aspect of our approach lies in the fact that the sentence embeddings generated through contrastive learning lead to a better separation of classes in an observable vector space, compared to embeddings generated through conventional learning for hate speech classification.

1

Introduction

1.1 Motivation

Au cours de ces dernières décennies, la croissance phénoménale de l'utilisation des plateformes en ligne, comme Twitter, Meta, YouTube, etc., a considérablement fait évoluer notre façon de communiquer. En effet, les réseaux sociaux ont permis de connecter des personnes du monde entier entre elles. De plus, ils offrent la possibilité de partager des opinions ou des informations. Bien qu'ils présentent de nombreux avantages, certaines personnes les utilisent pour diffuser et propager des contenus néfastes pour les sociétés, tels que les fausses informations, les discours haineux, etc.

Le Comité des Ministres du Conseil de l'Europe définisse un discours de haine comme "*tout type de communication qui incite à, promeut, diffuse ou justifie la violence, la haine ou la discrimination à l'encontre d'une personne ou d'un groupe de personnes en raison de leurs caractéristiques personnelles, telles que la race, la couleur, la religion, l'origine ethnique, etc.*". Jurgens et al. (2019) suggèrent qu'environ 40% des utilisateurs en ligne ont été victimes, à un moment donné, de tels discours. Les discours de haine peuvent avoir de graves conséquences sur leurs victimes en entraînant des dommages psychologiques, sociaux et pouvant aller jusqu'à des agressions physiques.

Confronté à ce constat, la détection des discours de haine est devenue une préoccupation importante pour la société. Récemment, un "code de conduite" [1] a été promulgué par la Commission de l'Union Européenne demandant aux grands groupes de l'internet (Twitter [2], MeTa, YouTube, Microsoft, etc.) de s'engager pour lutter contre les discours de haine en ligne. Ce "code de conduite" exige une action rapide des entreprises pour examiner le contenu signalé par leurs utilisateurs dans un délai de 24 heures. Le nombre exponentiel de contenus postés chaque jour sur les réseaux sociaux rend la modération très difficile, voire impossible. Il est estimé à 5900 [3] messages (tweets) postés par seconde sur Twitter, soit environ 500 millions de tweets par jour. C'est pourquoi une méthode automatique est nécessaire pour supprimer ses contenus des réseaux sociaux.

Les avancés en traitement automatique des langues (TAL) ont permis de développer des approches pour la détection des discours de haine. Les premières approches étaient fondées sur l'utilisation de classifieurs, tels que des machines à vecteur de support (SVM pour *Support Vector*

1. https://commission.europa.eu/strategy-and-policy/policies/justice-and-fundamental-rights/combatting-discrimination/racism-and-xenophobia/eu-code-conduct-countering-illegal-hate-speech-online_en
2. Depuis juillet 2023, Twitter est devenu X.
3. https://www.planetoscope.com/Internet-/1547-nombre-de-tweets-expedies-sur-twitter.html

Machine en anglais) ou des régressions logistiques (LR pour *logistic regression* en anglais) avec des caractéristiques sélectionnées manuellement, telles que des n-grammes de mots ou de caractères, le nombre de termes haineux, la polarité des mots ou des phrases, etc. (Nobata et al., 2016a; Davidson et al., 2017). Ensuite, les réseaux neuronaux ont fait émerger des modèles de langage qui sont génériques et qui sont appris sur de grands ensembles de données. Nous pouvons citer parmi d'autres les plongements de mots (Mikolov et al., 2013; Bojanowski et al., 2017; Pennington et al., 2014), les plongements de phrases (Cer et al., 2018; Conneau et al., 2017). De plus, les réseaux de neurones sont rapidement devenus des solutions efficaces pour détecter les discours de haine (Badjatiya et al., 2017). Avec l'évolution des architectures neuronales, notamment avec l'émergence des mécanismes d'attention (Bahdanau et al., 2014; Yin et al., 2016), de nouveaux modèles de langage se sont distingués. C'est le cas des modèles de langage pré-entraînés fondés sur les *transformers* (Vaswani et al., 2017a), tels que BERT (Devlin et al., 2019) ou RoBERTa (Liu et al., 2019a). Le plus de ces modèles de langage est qu'ils permettent d'affiner leurs poids sur des tâches spécifiques (*fine-tuning* en anglais), contrairement aux approches précédentes. Ces nouvelles approches ont permis l'élaboration de modèles de langage spécifiques à la haine comme HateBERT (Caselli et al., 2021), AngryBERT (Awal et al., 2021a) ou bien fBERT (Sarkar et al., 2021), et aussi spécialisés dans les réseaux sociaux comme BERTweet (Nguyen et al., 2020).

1.2 Objectifs de la thèse

L'objectif principal de cette thèse est de **proposer des approches automatiques pour détecter les discours de haine sur le réseau social de Twitter**. Ce réseau social est très utilisé dans le monde, notamment en France comptant en moyenne plus de cinq millions d'utilisateurs uniques chaque jour [4]. Beaucoup de chercheurs ont étudié la détection des discours de haine dans les tweets, ce qui a permis le développement de jeux de données de références pour notre tâche. Cependant, il est complexe de détecter les discours de haine dans les tweets car ils peuvent contenir un langage non standard avec des syntaxes incorrectes, des erreurs grammaticales, des abréviations, etc.

Nous nous sommes intéressés à deux axes de recherche pour la détection des discours de haine, à savoir l'intégration de différentes caractéristiques dans des réseaux neuronaux et l'étude de différentes méthodes d'apprentissage.

1.2.1 Intégration de différentes caractéristiques dans des réseaux neuronaux

Le premier axe de recherche se focalise sur l'intégration de différentes caractéristiques dans des réseaux neuronaux. Comme mentionné précédemment, les méthodes de détection des discours de haine ont beaucoup évoluées. Cependant, certaines caractéristiques linguistiques, tels que des caractéristiques de surfaces (nombre de mots haineux, ponctuation, casse des mots, etc.), des caractéristiques provenant de l'analyse de sentiments (polarité des mots ou des phrases) et des caractéristiques syntaxiques (partie du discours, analyse de dépendance), se sont révélées être efficaces pour détecter les discours de haine (Gitari et al., 2015; Nobata et al., 2016a). Cependant, ces caractéristiques ont majoritairement été utilisé dans des classifieurs traditionnels comme des SVM, LR, etc. (Chen et al., 2012; Burnap and Williams, 2015). Les plongements de phrases se sont également montré efficaces pour notre tâche (Djuric et al., 2015; Indurthi et al., 2019).

Notre objectif est de savoir quelles caractéristiques en complément des plongements de phrases sont utiles pour détecter les discours de haine. Nous étudions l'intégration de caractéristiques

4. `https://www.blogdumoderateur.com/chiffres-twitter/`

linguistiques pour notre tâche. Les caractéristiques linguistiques sont : les mots à caractères haineux, la ponctuation, la casse des mots, les émojis et les parties du discours. En effet, nous pensons que ces caractéristiques sont importantes pour détecter les discours de haine. Les mots haineux peuvent être des indicateurs de messages haineux. La ponctuation et la casse des mots peuvent être utilisées pour nuancer des émotions. Les émojis sont de petits symboles picturaux qui peuvent être utilisés pour transmettre des émotions, des idées, des concepts dans les messages et ils sont notamment très présents dans les tweets. Les parties du discours sont des catégories linguistiques qui classifient les mots en fonction de leur rôle grammatical et sémantique dans une phrase. De part le fait que les tweets peuvent contenir des erreurs syntaxiques et grammaticales, il semble intéressant d'utiliser les parties du discours pour identifier les messages haineux.

Nous nous sommes également focalisés sur l'utilité d'une caractéristique peu étudiée pour notre tâche, à savoir les expressions polylexicales (EP). Une EP est définie par des unités lexicales qui peuvent être décomposées en plusieurs lexèmes, et véhicule une idiosyncrasie lexicale, morphologique, sémantique, syntaxique et/ou statistique (Baldwin and Kim, 2010). En outre, une EP véhicule un sens différent du sens littéral des mots qui la compose. Les EP semblent intéressantes pour aider un système neuronal à détecter les discours de haine. De plus, des chercheurs soutiennent que les EP sont importantes pour la détection des discours de haine (Waseem et al., 2018; Stanković et al., 2020). Cependant, les jeux de données pour notre tâche ne sont pas annotés en termes d'EP. Donc, un de nos objectifs est d'étudier également la robustesse de systèmes d'identification d'EP dans les tweets. Ensuite, nous proposons différentes façons d'intégrer les EP dans des réseaux neuronaux fondés sur des plongements de phrases. Notre objectif est de répondre à plusieurs questions de recherche :

— L'intégration des EP dans un système neuronal est-elle utile pour la détection des discours de haine ?

— Quelles caractéristiques linguistiques, en complément des plongements de phrases, sont pertinentes pour la détection des discours de haine ?

— Comment intégrer efficacement les EP dans des systèmes neuronaux pour notre tâche ?

— Quel système d'identification d'EP est le plus performant pour la détection des discours de haine ?

1.2.2 Méthodes d'apprentissage pour la détection des discours de haine

Dans notre deuxième axe de recherche, nous nous focalisons sur deux méthodes d'apprentissage : multi-tâches et par contraste. Comme mentionné plus tôt, les réseaux de neurones ont surpassé l'utilisation de classifieurs tels que les SVM, les classifieurs de régression logistique, etc. Cependant, il existe différentes méthodes d'apprentissages ainsi que différentes architectures neuronales pour détecter les discours de haine. Une méthode d'apprentissage consiste à entraîner un réseau neuronal uniquement sur la tâche de détection des discours de haine (Badjatiya et al., 2017; Zhang et al., 2018; Badri et al., 2022). Une autre méthode d'apprentissage consiste à entraîner un modèle neuronal sur différentes tâches simultanément. On parle alors d'apprentissage multi-tâches (*multi-task learning* en anglais). Des chercheurs ont montré que l'utilisation de cette méthode améliore les performances de la détection des discours de haine en utilisant des tâches liées à la haine (Kapil and Ekbal, 2020; Rajamanickam et al., 2020; Zhou et al., 2021; Awal et al., 2021b; D'Sa et al., 2022). L'apprentissage multi-tâches peut permettre de partager des connaissances pour des tâches connexes (Kapil and Ekbal, 2020; Rajamanickam et al., 2020). De plus, certaines tâches peuvent avoir des ensembles de données plus petits ou plus rares. En utilisant l'apprentissage multi-tâches, les informations partagées entre ces tâches peuvent aider

à améliorer les performances des tâches moins riches en données, comme l'a montré D'Sa et al. (2022) pour la détection des discours de haine. En apprenant plusieurs tâches en même temps, le modèle peut capturer des caractéristiques communes entre les tâches. Cela peut conduire à une meilleure généralisation, où le modèle est capable d'obtenir de meilleures performances sur de nouvelles données (Awal et al., 2021a). L'apprentissage multi-tâches peut agir comme une forme de régularisation en limitant la complexité du modèle. Les tâches connexes peuvent encourager le modèle à apprendre des caractéristiques utiles et à éviter le sur-ajustement. En effet, des chercheurs ont montré qu'utiliser une architecture simple avec des mécanismes d'attention pouvait améliorer les performances de la détection des discours de haine (Safi Samghabadi et al., 2020; Zhou et al., 2021). Un de nos objectifs dans cette thèse est d'étudier l'apprentissage multi-tâches, avec pour tâche auxiliaire l'identification des EP. Pour rappel, certains chercheurs soutiennent que les EP sont importantes pour détecter les discours de haine. Nous pensons qu'un modèle basé sur un mécanisme d'attention permettra d'apprendre des caractéristiques importantes pour aider à améliorer la détection de la parole haineuse. Nous avons pour objectif de répondre à la question suivante :

— Est-il efficace d'utiliser comme tâche secondaire l'identification des EP pour détecter les discours de haine dans un système multi-tâches ?

— Est-il important d'incorporer un mécanisme d'attention pour apprendre à faire attention aux EP et aux discours de haine simultanément ?

Encore une méthode performante d'apprentissage est l'apprentissage par contraste (*contrastive learning* en anglais). L'objectif principal de l'apprentissage par contraste est de maximiser la similarité entre des exemples positifs (échantillons de la même classe ou similaires) et de minimiser la similarité entre des exemples négatifs (échantillons de classes différentes ou dissimilaires). Cet apprentissage a émergé dans le domaine de la vision par ordinateur (Jaiswal et al., 2020). Cette méthode d'apprentissage a récemment été étudié dans le domaine du TAL (Gao et al., 2021; Rethmeier and Augenstein, 2023). Cependant, très peu de recherches ont été effectuées pour la détection des discours de haine (Kim et al., 2022; Shapiro et al., 2022). L'idée principale de l'utilisation de cet apprentissage pour détecter les discours de haine est d'apprendre des représentations vectorielles des phrases afin qu'elles soient séparées dans l'espace vectoriel en fonction des classes. L'intuition est que cette séparation de vecteurs en fonction des classes facilite et améliore la classification des discours de haine. Dans cette thèse, nous proposons une étude préliminaire sur l'utilisation de l'apprentissage par contraste pour la détection des discours de haine. Nous avons pour but de répondre aux questions suivantes :

— Est-il efficace d'appliquer l'apprentissage par contraste pour la détection des discours de haine ?

— Comment créer les paires de tweets pour cet apprentissage ?

1.3 Contributions de la thèse

Nous décrivons ci-dessous nos contributions.

(1) Nous proposons d'intégrer différentes caractéristiques, telles que la casse des mots, les mots contenus dans un dictionnaire de termes haineux, la ponctuation, les parties du discours et les émojis, dans un système neuronal basé sur les plongements de phrases. Nous nous intéressons à l'apport de ces caractéristiques en supplément des plongements de phrases. Pour cela, nous proposons un système neuronal intégrant ces caractéristiques.

(2) Nous étudions la robustesse des systèmes d'identification d'EP dans des messages de Twitter. Pour cela, nous utilisons des systèmes fondés sur des méthodes différentes pour identifier les EP : l'une utilisant des règles et un dictionnaire et l'autre fondée sur les réseaux neuronaux. Nous proposons également une approche en deux étapes pour cette tâche. Il est important de noter que l'identification des EP dans les tweets a été très peu étudiée (Ramisch et al., 2023).

(3) Nous étudions l'impact des EP pour la détection des discours haineux. Nous utilisons une méthode automatique pour identifier les EP dans les tweets. Nous les intégrons directement en entrée de systèmes neuronaux afin de montrer leurs utilités pour la détection des discours de haine. Pour cela, nous proposons des systèmes neuronaux fondés sur des plongements de phrases intégrant les EP de différentes façons.

(4) Nous explorons l'apprentissage multi-tâches. Notre système est développé pour faire attention aux EP en parallèle de l'attention porté aux discours haineux. Nous utilisons en tâche supplémentaire l'identification des EP qui est une tâche syntaxique et sémantique.

(5) Nous étudions l'apprentissage par contraste. Nous proposons différentes méthodes pour créer des paires de tweets afin d'utiliser l'apprentissage par contraste. Nous proposons également différentes méthodes de décision basées sur la distance cosinus pour prédire les classes.

1.4 Liste des publications

Les travaux présentés dans cette thèse ont été publiés dans les articles suivants :

1. **Nicolas Zampieri**, Irina Illina, Dominique Fohr. Multiword expression features for automatic hate speech detection. In *Proceedings of the 26th International Conference on Natural Language & Information Systems* (NLDB), 8 pages, 2021.
2. **Nicolas Zampieri**, Carlos Ramisch, Irina Illina, Dominique Fohr. Identification of multiword expressions in tweets for hate speech detection. In *Proceedings of the 13th Language Resources and Evaluation Conference* (LREC), 8 pages, 2022.
3. **Nicolas Zampieri**, Carlos Ramisch, Irina Illina, Dominique Fohr. Identification des expressions polylexicales dans les tweets. In *Actes de la 29e Conférence sur le Traitement Automatique des Langues Naturelles* (TALN), 8 pages, 2022.
4. **Nicolas Zampieri**, Irina Illina, Dominique Fohr. Improving hate speech detection with self-attention mechanism and multi-task learning. In *Proceedings of 10th Language & Technology Conference* (LTC), 5 pages, 2023.

1.5 Structure de la thèse

Cette thèse est organisée de la façon suivante :

Le **chapitre 2** décrit l'état de l'art de la détection des discours haineux.

Le **chapitre 3** décrit les jeux de données, les pré-traitements et les mesures d'évaluation utilisés dans cette thèse.

Le **chapitre 4** présente l'étude de l'intégration de caractéristiques linguistiques en supplément de représentation des tweets. Plus précisément, nous examinons l'incorporation de caractéristiques telles que la casse des mots, les émojis, les mots présents dans un dictionnaire de termes haineux et la ponctuation.

Le **chapitre 5** concerne l'étude de la robustesse de différents systèmes d'identification d'EP pour détecter automatiquement les EP dans les tweets.

Le **chapitre 6** présente notre étude de l'intégration des EP dans la tâche de la détection des discours haineux. Nous développons deux réseaux de neurones qui s'appuient sur les plongements de phrases et intègrent les EP.

Le **chapitre 7** traite l'apprentissage multi-tâches utilisant l'auto-attention multi-têtes. Nous proposons un réseau neuronal basé sur l'attention multi-têtes pour apprendre simultanément deux tâches : la détection des discours haineux et l'identification des EP.

Le **chapitre 8** concerne l'apprentissage par contraste. Notre approche consiste à appliquer cet apprentissage de manière supervisée. Nous proposons différentes méthodes pour créer des paires de tweets d'apprentissage.

Le **chapitre 9** conclu la thèse en résumant les résultats obtenus. Nous proposons également des perspectives pour les futures recherches.

2

La détection des discours haineux : un état de l'art

La détection des discours haineux suscite un intérêt croissant ces dernières années. La détection automatique des discours haineux consiste à développer des méthodes et des modèles informatiques capables d'identifier et de classer les messages contenant des propos haineux, offensants ou discriminatoires. Cette tâche présente de nombreux défis, notamment en raison de la nature complexe et subjective des discours haineux, ainsi que de la diversité des formes qu'ils peuvent prendre.

Il n'existe pas de définition précise pour définir les discours de haine, ce qui rend la tâche d'annotation des données et l'évaluation des modèles complexes. Les définitions peuvent varier selon les contextes culturels, juridiques et sociétaux, et il est essentiel de tenir compte de ces nuances lors du développement des systèmes de détection.

Nous étudions dans un premier temps les définitions existantes des discours haineux (section 2.1). Puis, nous présentons les difficultés de la tâche de la détection des discours haineux (section 2.2. Ensuite, nous décrivons l'évolution des méthodes de détection des discours de haine (section 2.3).

2.1 Définitions des discours haineux

Il existe une multitude de définitions des discours de haine. La table 2.1 présente les définitions utilisées par des organismes gouvernementaux (Nations unies [5] et le conseil de l'Europe [6]), les réseaux sociaux (Twitter [7], YouTube [8] et Meta [9]) et les chercheurs (Nockeby, 2000; Nobata et al., 2016a; Davidson et al., 2017; Caselli et al., 2020). Malgré les différences entre ces définitions, elles s'accordent sur le fait que les discours de haine sont extrêmement néfastes et cibles des individus ou des groupes d'individus en se basant sur des caractéristiques (genre, origine ethnique, orientation sexuelle, etc.). Ces définitions ne sont pas précises, ce qui pose un énorme problème pour l'annotation de jeux de données. Par ailleurs, les chercheurs s'accordent sur le fait que les discours de haine sont une sous-catégories du langage abusif, dissociant les discours offensifs et

5. https://www.un.org/en/genocideprevention/documents/advising-and-mobilizing/Action_plan_on_hate_speech_FR.pdf
6. https://www.coe.int/fr/web/freedom-expression/hate-speech
7. https://help.twitter.com/fr/rules-and-policies/hateful-conduct-policy
8. https://support.google.com/youtube/answer/2801939?hl$=$fr
9. https://transparency.fb.com/fr-fr/policies/community-standards/hate-speech/

abusifs, des discours de haine (Stanković et al., 2020; Caselli et al., 2020; Yin and Zubiaga, 2021).

La détection automatique des discours de haine peut être considérée comme une tâche de classification de texte, c'est-à-dire la tâche vise à étiqueter un message comme étant haineux ou non. Pour entraîner des modèles d'apprentissage automatique, il est essentiel d'avoir des jeux de données annotées de manière fiable (Ross et al., 2016). Les annotations des jeux de données en termes de haine sont effectuées manuellement. Cependant, il est nécessaire de fournir des instructions précises aux annotateurs afin d'obtenir des annotations fiables.

Sources	**Définitions**
Organisation des Nations unies	"Tout type de communication, qu'il s'agisse d'expression orale ou écrite ou de comportement, constituant une atteinte ou utilisant un langage péjoratif ou discriminatoire à l'égard d'une personne ou d'un groupe en raison de leur identité, en d'autres termes, de l'appartenance religieuse, de l'origine ethnique, de la nationalité, de la race, de la couleur de peau, de l'ascendance, du genre ou d'autres facteurs constitutifs de l'identité."
Comité des Ministres du Conseil de l'Europe	"Tout type d'expression qui incite à, promeut, diffuse ou justifie la violence, la haine ou la discrimination à l'encontre d'une personne ou d'un groupe de personnes, ou qui les dénigre, en raison de leurs caractéristiques personnelles ou de leur statut réels ou attribués telles que la race, la couleur, la langue, la religion, la nationalité, l'origine nationale ou ethnique, l'âge, le handicap, le sexe, l'identité de genre et l'orientation sexuelle."
Twitter	"Il est interdit d'attaquer directement d'autres personnes en vous fondant sur la race, l'origine ethnique, l'origine nationale, la caste, l'orientation sexuelle, le sexe, l'identité sexuelle, l'appartenance religieuse, l'âge, un handicap ou toute maladie grave."
YouTube	"L'incitation à la haine est interdite sur YouTube. Nous supprimons tout contenu incitant à la violence ou à la haine contre des individus ou des groupes d'individus en fonction des caractéristiques suivantes : âge, caste, handicap, origine ethnique, identité et expression de genre, nationalité, groupe ethnique, statut d'immigration, religion, sexe/genre, orientation sexuelle, statut de victime d'un événement violent majeur ou de proche d'une victime, statut d'ancien combattant."
Meta	"Attaque directe contre des personnes, plutôt que contre des concepts ou des institutions, fondée sur ce que nous appelons des caractéristiques protégées : l'origine ethnique, l'origine nationale, le handicap, la religion, la caste, l'orientation sexuelle, le sexe, l'identité de genre, et les maladies graves. Nous définissons une attaque comme un discours violent ou déshumanisant, des stéréotypes offensants, une déclaration d'infériorité, une expression de mépris, de dégoût ou de renvoi, une insulte ou un appel à l'exclusion ou à la ségrégation..."
Nockeby (2000)	"Toute communication qui dénigre une personne ou un groupe en raison de certaines caractéristiques telles que la race, la couleur, l'origine ethnique, le genre, l'orientation sexuelle, la nationalité, la religion ou toute autre caractéristique." (traduit de l'anglais : "*Any communication that disparages a person or a group on the basis of some characteristic such as race, colour, ethnicity, gender, sexual orientation, nationality, religion, or other characteristic.*")

Nobata et al. (2016a)	"Un langage qui attaque ou dénigre un groupe en raison de sa race, de son origine ethnique, de sa religion, de son handicap, de son genre, de son âge, de son handicap ou de son orientation sexuelle/identité de genre." (traduit de l'anglais "*Language which attacks or demeans a group based on race, ethnic origin, religion, disability, gender, age, disability, or sexual orientation/gender identity.)*"
Davidson et al. (2017)	"Un langage utilisé pour exprimer de la haine envers un groupe ciblé ou qui est destiné à être insultant, humiliant ou insultant envers les membres du groupe." (traduit de l'anglais "*Language that is used to express hatred towards a targeted group or is intended to be derogatory, to humiliate, or to insult the members of the group.*")
Caselli et al. (2020)	"un langage blessant qu'un locuteur utilise pour insulter ou offenser un individu ou un groupe d'individus en se basant sur leurs qualités personnelles, leur apparence, leur statut social, leurs opinions, leurs déclarations ou leurs actions. Cela peut inclure des discours de haine, des langages péjoratifs, des jurons, des commentaires toxiques, des déclarations racistes et sexistes." (traduit de l'anglais : *Hurtful language that a speaker uses to insult or offend another individual or a group of individuals based on their personal qualities, appearance, social status, opinions, statements, or actions. This might include hate speech, derogatory language, profanity, toxic comments, racist and sexist statements.*)

Table 2.1 – Définitions des discours haineux.

2.2 Difficultés de la détection des discours haineux

Outre la difficulté d'annotation de jeu de données, il existe d'autres difficultés telles que le déséquilibre des classes et les biais, qui rendent la détection automatique des discours de haine complexe.

Les jeux de données ont souvent une distribution des classes déséquilibrées. En effet, les contenus haineux représentent une faible proportion des messages sur internet. Founta et al. (2018) estime que la proportion de contenus haineux est très faible sur Twitter (entre 0,1% et 3% des tweets). Cela rend difficile la création de jeux de données car les messages haineux sont comme une aiguille dans une botte de foin. Certains chercheurs ont utilisé des stratégies pour augmenter la présence des messages haineux lors de la collecte de données, comme l'utilisation de mots clés très présents dans les discours de haine (Waseem and Hovy, 2016; Davidson et al., 2017). Le déséquilibre entre les classes dans les jeux de données pose des problèmes aux classifieurs. Les classifieurs ont fortement tendance à prédire la classe majoritaire pour une instance de test. Par ailleurs, certains chercheurs ont abordé le problème du déséquilibre de classe au niveau des mesures d'évaluation des modèles (Bosco et al., 2018; MacAvaney et al., 2019; Mandl et al., 2021). En effet, la mesure d'évaluation *macro-F1* donne une importance équivalente entre toutes les classes contrairement à la *micro-F1* qui favorise la classe majoritaire (Sokolova and Lapalme, 2009; Narasimhan et al., 2016). La mesure d'évaluation *macro-F1* donne une meilleure estimation de la capacité de généralisation des classifieurs. Dans cette thèse, nous avons donc fait le choix d'utiliser la macro-F1 pour reporter les performances des modèles.

La présence de biais involontairement introduits dans les jeux de données pose aussi des problèmes pour l'apprentissage de modèles automatiques pour la détection des discours de haine. Ces biais peuvent provenir de différentes causes telles que la stratégie de l'échantillonnage, la

démographie des annotateurs, la période de collecte des données, etc. (Wiegand et al., 2019; Razo and Kübler, 2020). Ces biais dans les jeux de données peuvent réduire les performances des classifieurs de discours de haine.

Le biais d'échantillonnage fait référence au biais introduit dans un ensemble de données en raison de la manière dont il est collecté ou échantillonné. Ce biais peut survenir lorsque le processus de collecte inclut de manière déséquilibrés certains types de contenus. Pour la création de jeux de données, deux stratégies ont été proposé : échantillonnage basé sur des mots clés et échantillonage aléatoire renforcée (*boosted random sampling* en anglais). Un échantillonnage totalement aléatoire n'a pas un grand intérêt dû au fait de la faible proportion de discours de haine. L'échantillonnage à base de mots clés consiste à utiliser des termes sélectionnés manuellement pour collecter des données. Par exemple, Davidson et al. (2017) ont adopté cette stratégie d'échantillonnage basée sur des mots-clés avec le dictionnaire de termes haineux provenant de *hatebase.org* pour collecter des données. Similairement, Waseem and Hovy (2016) ont collecté leur jeu de données en utilisant des mots clés qui apparaissaient souvent dans les messages sexistes et racistes. Cependant, cette stratégie augmente le biais d'échantillonnage en raison de la présence disproportionné de termes non-corrélés avec la haine. Par exemple dans le jeu de données de Waseem and Hovy (2016), les termes les plus présent dans les discours de haine sont des mots non-haineux ("commentateur", "humoriste", "football", etc.) (Wiegand et al., 2019).

Le biais d'annotateur se produit lors de l'étiquetage des données par des annotateurs humains. Lorsque plusieurs annotateurs sont impliqués, il peut y avoir des divergences dans la manière dont ils interprètent et étiquettent les données. Certaines caractéristiques, telles que le genre, l'ethnicité, le niveau d'expertise, etc., d'un annotateur peuvent jouer un rôle important dans l'annotation des jeux de données. Des études ont été menées sur l'influence de ce biais pour l'annotation de jeux de données ainsi que sur les performances des classifieurs (Waseem, 2016; Breitfeller et al., 2019; Al Kuwatly et al., 2020; Wich et al., 2020; Waseem et al., 2021; Sap et al., 2022).

Le biais d'auteur se produit lorsqu'il y a une distribution déséquilibrée des messages provenant de différents auteurs dans un jeu de données. Cela signifie que certains auteurs contribuent de manière disproportionnée au jeu de données, tandis que d'autres auteurs sont sous-représentés. (Wiegand et al., 2019) ont montré que le jeu de données de Waseem and Hovy (2016) contenait un biais d'auteur : plus de 70% des messages haineux (sexiste ou raciste) provenaient seulement de trois auteurs.

Waseem et al. (2021) soutiennent qu'il est presque impossible de faire disparaître tous les biais dans les jeux de données et suggèrent que les chercheurs soient conscients du fait que les approches d'atténuation des biais ne peuvent pas réduire tous les biais existant dans les jeux de données.

Dans cette thèse, nous n'abordons pas les problématiques cités ci-dessus. Néanmoins, nous avons conscience que les jeux de données utilisés dans cette thèse sont déséquilibrés et contiennent des biais (d'auteur et d'échantillonage) pouvant poser des problèmes aux classifieurs.

2.3 Évolution des méthodes de la détection des discours de haine

2.3.1 Caractéristiques linguistiques

Ici, nous présentons les caractéristiques linguistiques étudiées pour la détection des discours de haine. Dans cette thèse nous considérons les caractéristiques de surfaces (comme les n-grammes de mots, la casse des mots, la ponctuation, etc.), syntaxiques (telles que les parties du discours,

les dépendances entre les mots, etc.) et celles issus de ressources lexicales (dictionnaire de mots de politesse, de mots haineux, etc.) comme étant des caractéristiques linguistiques.

De nombreux chercheurs utilisent la présence de mots injurieux, insultant, etc., en entrée de classifieurs. Razavi et al. (2010) ont créé un dictionnaire rassemblant des mots, des phrases et des expressions abusives et leur ont attribué des poids différents en fonction de leur impact potentiel. Chen et al. (2012) ont également construit un dictionnaire de mots abusifs, puis ont ajusté les scores offensifs associés à ces mots en fonction de leur contexte à l'aide d'analyse de dépendance (*dependency parsing* en anglais). Ces scores ont ensuite été combinés pour obtenir la valeur offensive d'une phrase afin de détecter finalement le contenu haineux. Gitari et al. (2015) ont utilisé une méthode basée sur des règles pour créer un dictionnaire permettant de détecter les discours haineux. Ils ont utilisé un analyseur de parties du discours et des dictionnaires de sentiments pour détecter les phrases subjectives. À partir de ces phrases subjectives, les auteurs ont dérivé trois ensembles de caractéristiques : des mots avec une polarité négative, des mots haineux et des modèles grammaticaux utilisant des thématiques pour effectuer une classification basée sur des règles. Nobata et al. (2016b) ont également utilisé la polarité des mots à l'aide de dictionnaires dans leur étude. Ils ont utilisé à la fois le nombre de mots de politesses et haineux.

Les caractéristiques de surface, telles que les n-grammes de mots, de caractères, ont beaucoup été étudiées pour la détection des discours de haine (Chen et al., 2012; Gitari et al., 2015; Nobata et al., 2016b; Burnap and Williams, 2015). Cependant, bien que ces caractéristiques soient prédictives, elles ne sont pas auto-suffisantes pour détecter tous les types de discours de haine. Nobata et al. (2016b) ont utilisé différentes caractéristiques telles que les n-grammes de caractères et de mots, le nombre de mots dans les commentaires, le nombre de signes de ponctuation, le nombre de mots haineux et de politesses, des caractéristiques syntaxiques telles que les étiquettes de parties du discours, les relations de dépendance dérivées à l'aide de l'analyse de dépendance, etc. La combinaison de toutes les caractéristiques étudiées a donné de meilleures performances pour la détection des discours haineux. Burnap and Williams (2015) ont couplé l'utilisation des relations de dépendance entre les mots, les termes haineux et les n-grammes de mots. Ils ont montré que l'utilisation de cette combinaison de caractéristiques améliorait significativement les résultats comparativement à l'utilisation indépendante de ces caractéristiques. Comme Burnap and Williams (2015), Davidson et al. (2017) ont utilisé différentes caractéristiques, telles que les étiquettes de parties du discours, les n-grammes, les scores de sentiment pour chaque commentaire à l'aide d'un dictionnaire de sentiment, des indicateurs pour les hashtags, les mentions, les retweets [10], le nombre de caractères, de mots et de syllabes, etc.

Des caractéristiques issues de l'analyse de sentiment (*sentiment analysis* en anglais), telles que la polarité des mots et des phrases, ont aussi été utilisées pour la détection des discours de haine (Sood et al., 2012; Burnap et al., 2015; Njagi et al., 2015; Van Hee et al., 2015, 2018). Njagi et al. (2015) ont utilisé un classifieur d'analyse de sentiment afin de donner une polarité aux phrases. Ils ont ensuite utilisé la polarité des phrases pour apprendre à détecter les messages haineux. Van Hee et al. (2015) ont utilisé la polarité des mots (positif, négatif ou neutre) comme caractéristique. Ils ont montré qu'utiliser cette caractéristique combinée avec des caractéristiques de surface telles que des n-grammes de mots améliorait les performances. Burnap et al. (2015) ont utilisé la sortie d'un analyseur de sentiment SentiStrength (Thelwall et al., 2010) en tant que caractéristique pour la détection de tension dans les messages. La sortie de l'analyseur SentiStrength permet de prendre en compte le degré de la polarité du sentiment en plus de donner une indication sur la polarité d'une phrase. Van Hee et al. (2018) ont utilisé plusieurs caractéristiques telles que les n-grammes de mots, les n-grammes de caractères, des indicateurs

10. Marqueur de partage des tweets

pour les noms propres, la polarité des mots, etc. Ils ont utilisé un dictionnaire de sentiment pour définir un score de polarité sur les mots. Ils ont montré que la combinaison des différentes caractéristiques étudiées améliorait les performances comparée à l'utilisation des n-grammes de mots seuls.

Dans cette thèse, nous nous intéressons à l'intégration de différentes caractéristiques, notamment des expressions polylexicales pour la détection des discours de haine.

Une expression polylexicale est définie par un groupe d'unités lexicales (deux lexèmes ou plus) qui ont un sens différent du sens littéral. Baldwin and Kim (2010) définissent les expressions polylexicales par des unités lexicales qui peuvent être décomposées en plusieurs lexèmes, et exposent une idiosyncrasie lexicale, morphologique, syntaxique, sémantique et/ou statistique. Les expressions polylexicales ont très peu été étudiées pour la détection des discours haineux. Cependant, quelques recherches ont été menées sur l'utilisation des expressions polylexicales pour la détection des discours haineux (Maisto et al., 2017; Waseem et al., 2018; Stanković et al., 2020).

Maisto et al. (2017) ont proposé un ensemble de règles et un dictionnaire pour annoter automatiquement des contenus sensibles concernant le cyber-harcèlement. Le dictionnaire est composé de termes haineux. Ce dictionnaire est utilisé avec les règles pour déterminer les expressions polylexicales liées aux discours de haine. Ils ont observé qu'utiliser les mots composants une expression polylexicale comme une seule unité lexicale permettait d'améliorer la détection des contenus non-désirables. Cependant, leur méthode d'évaluation n'est pas clairement définie. (Waseem et al., 2018) ont représenté les expressions polylexicales (de longueur de deux lexèmes) en concaténant les mots les composant. Par exemple, l'expression *take off* devient *take_ off* dans le texte. Ils ont collecté des expressions polylexicales en utilisant la méthode sac-de-mots pour extraire des bi-grammes de mots qui sont les plus fréquents. L'intuition des chercheurs est que les expressions polylexicales qui ont un sens distinct de leurs mots constitutifs peuvent être représentées de manière plus précise en tant que leurs propres représentations textuelles plutôt que d'avoir leur sens dilué par d'autres exemples d'entraînement. Stanković et al. (2020) ont utilisé ces expressions pour la détection des discours de haine dans la langue serbe. Ils ont ajouté des expressions polylexicales haineuses dans le dictionnaire de termes haineux HurtLex. Ils ont utilisé des mots déclencheurs pour collecter environ 4624 expressions en utilisant différentes sources, telles que des dictionnaires, collections de messages et résultats de recherches en ligne. Ensuite, ils ont manuellement vérifié la nature des expressions collectées et ils les ont annoté en trois catégories : abusive, potentiellement abusive et non abusive. Cette classification manuelle a été effectuée en cherchant les expressions dans des jeux de données provenant de Twitter, dans d'autres sources d'internet et de la littérature. Ils ont obtenu 1260 expressions abusives, 462 potentiellement abusives et 2902 non abusives. L'ajout des expressions polylexicales a été effectué manuellement. Ils soutiennent que les expressions polylexicales sont importantes pour détecter les discours de haine tout comme Waseem et al. (2018).

Il est important de noter que la majorité des études basées sur des caractéristiques utilise des classifieurs statistiques traditionnels, tels que des machines à vecteur de support (SVM pour *Support Vector Machine* en anglais) (Hearst et al., 1998), des classifieurs de régression logistique (LR pour *logistic Regression* (McCullagh and Nelder, 1989), bayésien naïf (*Naïve Bayes* en anglais) (Witten and Frank, 2002), arbre de décision (*Decision Trees* en anglais) (Hearst et al., 1998), etc., pour détecter les discours de haine (Razavi et al., 2010; Chen et al., 2012; Nobata et al., 2016a; Davidson et al., 2017).

Avec l'évolution des technologies dans le TAL, les réseaux de neurones ont rapidement surpassés l'utilisation de classifieurs traditionnels (Badjatiya et al., 2017). De plus, les nouvelles méthodes d'apprentissage basé sur les réseaux neuronaux ont fait émerger de nouvelles caracté-

ristiques, tels que les plongements de mots, les plongements de phrases, ainsi que les modèles de langage que nous décrivons dans les sections suivantes.

2.3.2 Plongements de mots

Un plongement (*embeddings* en anglais) est une représentation vectorielle d'un mot, d'une phrase ou d'un document dans un espace mathématique continu. Il permet de capturer les informations sémantiques et contextuelles du texte de manière compacte et structurée. Les plongements de mots ont vu leurs apparitions avec l'utilisation des réseaux neuronaux pour l'apprentissage profond. Un système d'apprentissage profond peut apprendre des représentations sémantiques et syntaxiques à partir du texte grâce à la sélection automatique de caractéristiques. Ces systèmes utilisent une structure à plusieurs couches pour apprendre des représentations abstraites à partir de l'entrée. Cependant, ces systèmes requièrent une grande quantité de données pour apprendre des représentations. Les plongements de mots pré-entraînés ont été très utilisés dans différentes tâches du TAL. C'est le cas par exemple des plongements de mots *word2vec* (Mikolov et al., 2013), FastText (Bojanowski et al., 2017), et GloVe (Pennington et al., 2014), qui sont appris afin que des mots sémantiquement similaires soient proches dans un espace vectoriel de grande dimension. Mikolov et al. (2013) ont proposé d'apprendre les plongements de mots de *word2vec* en suivant deux stratégies d'apprentissage : saut de mots (*Skip-Gram* en anglais) et sac de mots continus (CBOW pour *Continuous Bag of Words* en anglais). L'objectif de l'apprentissage par saut de mots est de prédire les mots voisins étant donné un mot d'entrée. A l'inverse, la méthode d'apprentissage CBOW consiste à prédire un mot étant donné ses mots environnants. Bojanowski et al. (2017) ont proposé une extension des plongements *word2vec*, en proposant d'enrichir les plongements de mots par la représentations de sous-mots. Le principe fondamental de FastText est de considérer chaque mot comme une collection de sous-mots (ou n-grammes de caractères). Cette représentation permet à FastText de capturer les informations de caractères et les affixes qui sont essentiels pour les langues où les préfixes et suffixes jouent un rôle important dans la signification des mots. Le calcul des plongements de FastText utilise également une approche de type saut de mots. Cependant, la prédiction des mots voisins est effectuée non seulement pour les mots complets, mais aussi pour tous les sous-mots présents. Contrairement aux plongements de mots de *word2vec* et de FastText, les plongements de GloVe sont calculés à partir d'une matrice de co-occurrence qui enregistre la fréquence à laquelle les mots apparaissent ensemble dans le contexte d'une fenêtre donnée. Cette matrice de co-occurrence est ensuite utilisée pour apprendre les vecteurs de mots en minimisant une fonction de coût qui vise à optimiser les vecteurs de mots de manière à ce que leur produit scalaire soit proportionnel au logarithme de leur fréquence de co-occurrence. Cette approche permet de capturer les relations de similarité et de sémantique entre les mots, car les mots qui apparaissent fréquemment ensemble auront des vecteurs similaires.

Badjatiya et al. (2017) ont expérimenté l'utilisation de classifieurs traditionnels (SVM, régression logistique, etc.) avec des caractéristiques telles que les n-grammes de caractères, pondération Fréquence des termes - Inverse de la fréquence dans les documents (TF-IDF pour *Term Frequency - Inverse Document Frequency* en anglais), des sac de mots (*Bag-of-Words* en anglais) contenant les plongements de mots provenant de GloVe. Ils ont comparé ces approches avec des approches basées sur des réseaux neuronaux, tels que des réseaux de neurones convolutionnelles (CNN pour *Convolutional Neural Network* en anglais) (Kim, 2014), des réseaux neuronaux récurrent de mémoire à long et court terme (LSTM pour *Long Short-Term Memory* en anglais) (Hochreiter and Schmidhuber, 1997). Ces réseaux neuronaux ont été entraînés avec des plongements de mots initialisés aléatoirement et avec des plongements de mots pré-entraînés de GloVe et de FastText.

Badjatiya et al. (2017) ont démontré que les plongements de mots provenant de GloVe (utilisés dans un sac de mots) obtenaient de meilleurs performances que les n-grammes de caractères avec des classifieurs traditionnels. Ils ont également constaté que ces approches basées sur des réseaux neuronaux surpassaient les classifieurs statistiques n'utilisant pas de réseaux neuronaux. Des études similaires ont été réalisée par Gambäck and Sikdar (2017a) et Zhang et al. (2018). Gambäck and Sikdar (2017b) ont constaté qu'avec un réseaux de neurones CNN utilisant des plongements pré-entraînés (*word2vec*) donnait des résultats significativement meilleures comparé à un classifieur de LR utilisant des n-grammes de caractères et à l'utilisation de plongements appris en même temps que le modèle CNN. Zhang et al. (2018) ont proposé une architecture neuronale combinant des couches de CNN et de réseaux neuronaux récurrents GRU (pour *Gated Recurrent Unit* en anglais) (Cho et al., 2014a,b). Ils ont montré que leur proposition d'architecture CNN-GRU surpassait un classifieur utilisant différentes caractéristiques telles que les n-grammes, les parties du discours, la polarité de la phrase, etc. Ils ont également constaté que leur proposition d'architecture surpassait un modèle fondé uniquement sur des CNN. Badri et al. (2022) ont proposé de combiner plusieurs plongements de mots en entrée d'un réseau neuronal récurrent bi-directionnel. Pour représenter les mots, ils ont combiné les plongements de mots provenant de GloVe et de FastText. Ils ont montré une amélioration significative comparé à l'utilisation indépendante de ces plongements de mots. (Naseem et al., 2019) ont combiné la polarité des mots avec des plongements de mots contextuels de ELMo (Peters et al., 2018) et des plongements de mots provenant de *word2vec*. Ils ont montré que ces caractéristiques amélioraient les performances pour la détection des discours de haine sur trois jeux de données différents.

2.3.3 Plongements de phrases

(Djuric et al., 2015) ont proposé une autre approche qui consiste à utiliser des plongements qui représentent directement le texte entier pour la détection des discours de haine. Ces plongements de phrases (ou de document) comme *doc2vec* (Le and Mikolov, 2014) se sont révélés beaucoup plus efficaces que la simple moyenne des plongements de mots (Nobata et al., 2016b). Indurthi et al. (2019) ont exploré l'utilisation de plongements de phrase générés par des encodeurs de phrases, tels que InferSent (Conneau et al., 2017) ou bien *Universal Sentence Encoder* (USE) de Cer et al. (2018), etc. L'encodeur de phrases InferSent est basé sur des réseaux de neurones récurrents. Il utilise deux types d'architecture neuronale : un bidirectionnel LSTM et un arbre de LSTM (*Tree-LSTM* en anglais). L'apprentissage de InferSent se fait en utilisant un grand jeu de données de paires de phrases, où chaque paire est annotée comme étant sémantiquement équivalente ou non. Concernant l'encodeur USE, son architecture de base utilise des réseaux de neurones récurrent combinés à des mécanismes d'attention. Le processus d'encodage de USE consiste à prendre une phrase en entrée et à la passer à travers un réseau de neurones qui capture les relations entre les mots et les phrases dans le contexte global du texte. Un avantage clé de l'encodeur USE est qu'il est entraîné sur un jeu de données massif et diversifié, ce qui lui permet d'apprendre des représentations de phrases génériques. Indurthi et al. (2019) ont évalué différents plongements de phrases, tels que InferSent, USE, la concaténation par la moyenne de puissance des embeddings de mots (de l'anglais *Concatenated Power Mean Word Embedding*) (Rücklé et al., 2018), ELMo, etc., avec différents classifieurs statistiques (SVM, LR, etc.). Ils ont montré, lors de la campagne d'évaluation organisée par Basile et al. (2019), que les plongements de phrases générés par USE surpassaient les performances d'un modèle de langage, basé sur les *transformers* comme le modèle de langage de (Devlin et al., 2019), affinant ses poids (*fine-tuning* en anglais) lors de l'apprentissage.

Avec l'évolution des architectures neuronales utilisant un mécanisme d'attention, l'architec-

ture de l'encodeur USE a aussi évolué : l'architecture de réseaux neuronaux récurrent bidirectionnel a été remplacée pour une architecture de type *transformer*. Cette architecture neuronale a révolutionné le domaine du TAL, avec l'arrivée des modèles de langage pré-entraînés permettant d'être affinés lors de l'apprentissage sur une tâche particulière.

2.3.4 Caractéristiques extraites des modèles de langage pré-entraînés

Les *transformers* sont des couches neuronales basées sur de l'auto-attention multi-têtes (expliquée en section 7.1) proposée par Vaswani et al. (2017a). Ils ont permi l'élaboration de modèles de langages pré-entraînés performant comme les modèles de langues suivants : Représentations Bidirectionnelles d'Encodeurs basés sur des Transformers (BERT pour *Bidirectional Encoder Representations from Transformers* en anglais) de Devlin et al. (2019) ou RoBERTa (de l'anglais *Robustly Optimized BERT Approach*) de Liu et al. (2019b). Ces modèles de langages sont entraînés sur de vastes jeux de données non annotés. Par exemple, BERT est entraîné sur de grands jeux de données à l'aide de deux tâches : la tâche de modèle de langage masqué (MLM pour*masked language model* en anglais) et la tâche de détection de phrases adjacentes (NSP pour *Next Sentence Prediction* en anglais). Pour la tâche de MLM, les mots sont masqués de façon aléatoire. Pour la tâche de NSP, les paires de phrases sont également sélectionnées aléatoirement. Ces méthodes permettent à l'encodeur d'apprendre des relations sémantiques et syntaxiques pertinentes. Concernant RoBERTa, l'architecture est similaire à celle de BERT. La différence avec le modèle de langage BERT est l'apprentissage sur lequel est entraîné RoBERTa. Le modèle de langage RoBERTa est entraîné sur la tâche de MLM en utilisant un masquage des mots dynamiques où certains mots sont masqués à chaque itération, ce qui permet au modèle de voir plus de variations des données lors de l'apprentissage. De plus, RoBERTa utilise également une plus grande quantité de données d'apprentissage et des phrases plus longues.

De nombreux modèles de langages spécialisés, basés sur BERT ou RoBERTa, ont rapidement émergé pour différentes tâches du TAL. C'est le cas de la détection des discours de haine avec les modèles de langage HateBERT (Caselli et al., 2021) et fBERT (Sarkar et al., 2021). Ces deux modèles de langages sont basés sur l'architecture de BERT. Concernant le modèle de langage HateBERT, il est ré-entraîné sur la tâche de MLM en utilisant des commentaires de Reddit dont les conversations ont été supprimées du réseau social pour des raisons de contenus abusifs. Concernant le modèle de langage fBERT, il est ré-entraîné sur un grand nombre de tweets offensifs. On trouve aussi des modèles de langage spécifiques pour un réseau social, comme le modèle BERTweet (Nguyen et al., 2020). Le modèle de langage BERTweet est basé sur l'architecture neuronale de RoBERTa. Ce modèle est entraîné sur un grand nombre de tweets sur la tâche MLM comme RoBERTa. De plus, le modèle BERTweet utilise un pré-traitement différent des modèles BERT et RoBERTa dû à sa spécialisation dans le langage des tweets. Dans son pré-traitement, il prend en compte les émojis sous forme textuelle afin d'apprendre une représentation pour ceux-ci.

Il est important de noter que ces modèles de langages ont la particularité de pouvoir être facilement affinés pour différentes tâches en utilisant des jeux de données spécifiques. Cela permet aux modèles de langage d'apprendre des caractéristiques pertinentes en affinant leurs poids lors de l'apprentissage sur une tâche spécifique.

Cependant, on peut aussi extraire des plongements de mots ou de phrases de ces modèles de langues (ex : [CLS] dans BERT ou <s> dans BERTweet). Ils sont généralement appelés plongements contextuels. D'Sa et al. (2020) ont comparé les performances de l'utilisation des plongements de mots générés par FastText et le modèle de langage BERT en tant que caractéristiques pour les classifieurs fondés sur des réseaux neuronaux, tels que CNN et BiLSTM, avec l'affinage des poids du modèle de langage BERT. Ils ont constaté que l'affinage du modèle de

langage BERT obtenait des performances nettement supérieures. Pamungkas and Patti (2019) ont combiné un dictionnaire de termes abusifs appelé HurtLex avec des plongements contextuels de mots provenant du modèle de langage BERT. Ils ont constaté que la combinaison de ces deux caractéristiques améliorait les résultats avec un système basé sur des LSTM. Caselli et al. (2021) ont montré que leur modèle de langage HateBERT surpassait les performances de BERT sur différents jeux de données pour la détection des discours de haine. Il en est de même pour le modèle de langage fBERT, Sarkar et al. (2021) ont observé que leur modèle de langage obtenait de meilleures performances que les modèles de langage HateBERT et BERT. Nguyen et al. (2020) ont démontré que leur modèle de langage surpassait RoBERTa sur plusieurs tâches du TAL appliquées sur des tweets.

2.4 Méthodes d'apprentissage pour la détection des discours haineux

Dans cette section, nous décrivons des recherches utilisant différentes méthodes d'apprentissage profond pour la détection des discours haineux. Nous nous focalisons sur deux méthodes d'apprentissage : multi-tâches (section 2.4.2) et par contraste (section 2.4.3).

2.4.1 Apprentissage par transfert

L'apprentissage par transfert (*transfert learning* en anglais) est une approche du domaine de l'apprentissage automatique où les connaissances et les modèles acquis lors de l'apprentissage sur une tâche peuvent être utilisés pour améliorer les performances sur une autre tâche. Le processus d'apprentissage par transfert peut impliquer le partage de couches, de paramètres de modèles pré-entraînés d'une tâche source vers une tâche cible. L'idée fondamentale est que certaines structures ou représentations apprises sur la tâche source peuvent être utiles pour la tâche cible, même si les deux tâches sont distinctes. Le transfert d'apprentissage est couramment utilisé dans le TAL, notamment avec l'affinage des poids de modèles pré-entraînés tels que BERT ou RoBERTa. Il en est de même pour la détection des discours de haine (D'Sa et al., 2020; Sarkar et al., 2021; Caselli et al., 2021). L'apprentissage par transfert a aussi beaucoup été utilisé pour améliorer la généralisation des systèmes de détection de haine (Yin and Zubiaga, 2021; Bose, 2023).

Dans cette thèse, nous n'avons pas conduit de recherche sur l'application de l'apprentissage par transfert pour la détection des discours de haine.

2.4.2 Apprentissage multi-tâches

L'apprentissage multi-tâches (*multi-task learning* en anglais) est une approche en apprentissage automatique où un modèle est entraîné à effectuer plusieurs tâches différentes simultanément plutôt que de les traiter séparément. Le principe de base de l'apprentissage multi-tâches est d'utiliser des informations apprises lors de la résolution d'une tâche pour aider à améliorer les performances sur d'autres tâches connexes. Plusieurs chercheurs se sont intéressés à l'utilisation de cet apprentissage pour la détection des discours haineux (Kapil and Ekbal, 2020; Zhou et al., 2021; Awal et al., 2021b; Rajamanickam et al., 2020).

Kapil and Ekbal (2020) ont proposé une approche d'apprentissage multi-tâches utilisant plusieurs jeux de données de discours de haine. Ils ont expérimenté différentes architectures neuronales, telles que les CNN, les LTSM et la combinaisons des deux. Leurs approches utilisent la concaténation des plongements de mots et de caractères en entrée du modèle. Les couches neuronales (CNN, LSTM et CNN-LSTM) affinent leurs poids simultanément sur les différents jeux

de données d'entraînement. Seule les couches de classification sont propres à chacun des jeux de données. Le système neuronal apprend simultanément à classifier les messages des différents jeux de données en partageant les poids des couches neuronales. Ils ont montré qu'apprendre simultanément leur système sur cinq jeux de données améliorait les performances des résultats comparés à l'apprentissage sur moins de jeux de données. D'Sa et al. (2022) ont tiré des conclusions similaires à celles de Kapil and Ekbal (2020) dans une configuration avec de faibles ressources d'apprentissage. Comme Kapil and Ekbal (2020), Zhao et al. (2021) ont montré qu'apprendre un système neuronal simultanément sur différents types de discours de haine (agression, toxicité, attaque direct) améliorait les performances comparés à l'apprentissage sur chacune des tâches séparément.

Rajamanickam et al. (2020) suggèrent que les caractéristiques émotionnelles sont bénéfiques pour détecter les discours haineux. Ils ont proposé une approche multi-tâche en utilisant comme tâche secondaire la détection d'émotion. Ils ont montré que l'apprentissage multi-tâche permettait d'apprendre des caractéristiques provenant de la détection d'émotion pour aider à la tâche de détection des discours haineux. Zhou et al. (2021) ont utilisé l'analyse de sentiment comme tâche secondaire. Ils ont utilisé une architecture neuronale de type mélange d'experts (*mix of expert* en anglais) pour partager les caractéristiques des deux tâches. Les couches neuronales partagées entre les deux tâches sont trois modules neuronaux identiques fondés chacun sur de l'attention multi-têtes. Ils ont utilisé la concaténation des caractéristiques de plongements de mots provenant de *word2vec* avec des étiquettes pour les mots haineux. Ils ont montré que leur approche surpassé l'affinage des poids de BERT sur deux jeux de données. De plus, Chiril et al. (2022) ont étudié les caractéristiques émotionnelles extraites des ressources sémantiques pour l'analyse de sentiment Sentic (SenticNet, EmoSenticNet) (Cambria et al., 2020; Poria et al., 2013) et des dictionnaires de haine structurés (HurtLex). Ils ont mis en œuvre l'apprentissage multi-tâches et ont obtenu les meilleurs résultats avec des modèles qui utilisaient des données provenant de ces ressources sémantiques. En complément, Plaza-Del-Arco et al. (2021) ont soutenu l'hypothèse de la relation entre les tâches de discours de haine et les sentiments, les émotions ainsi que les cibles des discours, en utilisant une architecture neuronale d'apprentissage multi-tâches. De plus, en utilisant un modèle basé sur les *transformers*, del Arco et al. (2021) ont proposé un modèle multi-tâches qui exploite des connaissances partagées sur les sentiments et les émotions pour identifier les discours de haine dans les tweets en espagnol.

Awal et al. (2021b) ont proposé le modèle de langage AngryBERT qui apprend simultanément à détecter les discours haineux et les cibles en tant que tâches secondaires. Leur proposition de modèle de langage a montré des meilleures performances que d'autres modèles de langage tels que BERT ou RoBERTa. Safi Samghabadi et al. (2020) ont proposé d'ajouter une couche d'attention sur la sortie du modèle de langage BERT en utilisant un paradigme d'apprentissage multi-tâches. Leur approche a démontré des résultats prometteurs dans le contexte de la détection des actes de misogynie.

Dans cette thèse, nous nous intéressons à l'apprentissage multi-tâches sur les tâches d'identification des EP et la détection des discours de haine afin d'améliorer les performances sur cette dernière.

2.4.3 Apprentissage par contraste

L'apprentissage par contraste (*contrastive learning* en anglais) constitue une méthode d'apprentissage automatique qui vise à acquérir des représentations sémantiques en capitalisant sur les distinctions discernables entre des instances au sein d'un ensemble de données. L'objectif principal de l'apprentissage par contraste est de maximiser la similarité entre des exemples de

la même classe et de minimiser la similarité entre des exemples de classes différentes. Cette apprentissage a émergé dans le domaine de la vision par ordinateur pour la classification d'images (Hadsell et al., 2006; He et al., 2019). Les premières utilisations de l'apprentissage par contraste dans le domaine du TAL sont récentes (Rethmeier and Augenstein, 2023).

L'apprentissage par contraste a notamment été utilisé pour améliorer les performances pour la tâche de similarité sémantique textuelle (STS, pour *Semantic Textual Similarity* en anglais) qui vise à mesurer la similitude sémantique entre deux phrases (Reimers and Gurevych, 2019; Qi et al., 2022). C'est aussi le cas pour la tâche de relation entre deux phrases (très similaire à la tâche de STS) qui a pour objectif de classifier des paires de phrases en différentes catégories, par exemple *implication*, *neutre* ou *contradiction* (Gao et al., 2021). L'apprentissage par contraste a aussi été utilisé pour la génération de texte, ce qui a permis de réduire la répétition dans les textes générés ainsi que d'augmenter la fluidité de ceux-ci (Uehara et al., 2020). De plus, cet apprentissage a montré son efficacité pour la classification de texte (Pappas and Henderson, 2019; Chen et al., 2022). En ce qui concerne la détection des discours de haine, très peu de chercheurs ont utilisé l'apprentissage par contraste (Kim et al., 2022; Zhong et al., 2022; Shapiro et al., 2022).

Kim et al. (2022) ont proposé une approche basée sur l'apprentissage par contraste pour améliorer la détection des discours haineux implicites. Leur approche, appelé *ImpCon*, apprend à classifier les discours haineux et à rapprocher les différents messages de leurs implications. Pour cela, ils ont utilisé le jeu de données créé par ElSherief et al. (2021) qui contient l'annotation binaire (haine et non-haine) des messages, ainsi que des textes décrivant les conséquences de ces messages. Par exemple, "*People of color are inferior to white people*" est le texte décrivant l'implication du message "*my world orbits around whites as it should. laughable moment though.*". L'objectif de Kim et al. (2022) est de rapprocher les messages qui ont la même implication tout en apprenant à les classifier correctement. Ils ont montré que l'apprentissage par contraste, avec des modèles de langage comme BERT et HateBERT, généralise mieux que l'utilisation de l'apprentissage traditionnel pour la classification des discours de haine pour la prédiction entre différents jeux de données.

Zhong et al. (2022) ont étudié l'apprentissage par contraste pour détecter les discours haineux avec un modèle basé sur l'apprentissage multi-tâches. Pour créer les paires positives ils masquent les mots non-importants pour la classification des discours haineux, et à l'inverse ils masquent les mots importants pour créer les paires négatives. Ensuite ils apprennent au modèle à rapprocher les paires positives tout en éloignant les contenus des paires négatives. L'intuition est d'obtenir une meilleure représentation des contenus en se focalisant uniquement sur les mots importants. Cette intuition est venue du fait que des termes non-haineux peuvent être utilisés pour exprimer la haine, comme par exemple le terme "*pigs*" dans l'exemple haineux "*These guys are all a bunch of pigs*". A l'inverse, des termes péjoratifs ou insultants peuvent être utilisés dans des contenus normaux, comme dans l'exemple "*He's a fucking good player*".

Shapiro et al. (2022) ont proposé le système AlexU-AIC, qui utilise l'apprentissage par contraste pour la détection des discours haineux dans différents dialectes arabes. Leur approche consiste à créer des paires positives et négatives en mélangeant les différents dialectes arabes. Ils ont montré que le système basé sur l'apprentissage par contraste génère des représentations de phrases de meilleure qualité par rapport à un système utilisant uniquement l'apprentissage de la classification des discours haineux.

Nous proposons une étude préliminaire, réalisée en fin de thèse, sur l'utilisation de l'apprentissage par contraste pour la détection des discours haineux dans le chapitre 8.

2.5 Résumé du chapitre

Dans ce chapitre, nous avons fourni un aperçu de l'état actuel des connaissances sur la détection des discours haineux. Nous avons discuté des différentes définitions existantes du discours de haine. Nous avons également exposé quelques difficultés de la détection des discours de haine.

Ensuite, nous avons exploré différentes caractéristiques ainsi que l'évolution des classifieurs utilisés pour la détection des discours de haine. Nous avons observé que les caractéristiques de plongements surpassent les caractéristiques linguistiques. De plus, la combinaison de plongements (de mots ou de phrases) avec des caractéristiques sémantiques (provenant de l'analyse de sentiments) et syntaxiques (comme l'analyse de dépendances) montre des performances remarquables pour détecter les discours de haine. Cependant, les expressions polylexicales n'ont pas encore été étudiées en profondeur pour la détection des discours de haine.

Nous avons également exposé l'utilisation de différentes méthodes d'apprentissage profond, telles que l'apprentissage par transfert, multi-tâches et par contraste. Nous avons constaté que l'apprentissage multi-tâches a été souvent utilisé avec des tâches d'analyse de sentiments comme tâche secondaire pour améliorer la détection des discours de haine. Nous avons remarqué que peu de travaux ont été réalisé sur l'utilisation de l'apprentissage par contraste pour la détection des discours de haine. Cependant, l'utilisation de cet apprentissage a montré des performances similaires comparé à l'apprentissage de la classification des discours de haine.

3

Jeux de données, pré-traitements et mesures d'évaluation

Dans ce chapitre, nous présentons les jeux de données utilisés dans nos expériences afin de répondre à nos questions de recherches sur la détection de la parole haineuse et sur l'identification des EP dans les tweets. Dans un premier temps, nous décrivons les jeux de données annotées pour la détection automatique de la parole haineuse dans les tweets. Dans un second temps, nous présentons les jeux de données utilisés pour l'identification des EP dans les tweets. Dans chaque section, nous expliquons le pré-traitement appliqué aux jeux de données ainsi que les mesures d'évaluation utilisées.

3.1 Les jeux de données pour la détection de la parole haineuse

Il existe plusieurs jeux de données annotés en termes de haine. Afin de mener à bien nos expériences, nous avons décidé de nous focaliser sur quatre jeux de données de tweets en anglais : Waseem(Waseem and Hovy, 2016), SemEval 2019 task 5 (Basile et al., 2019), Davidson (Davidson et al., 2017) et Founta (Founta et al., 2018). Ils sont très utilisés dans le domaine de la détection de la parole haineuse. Les jeux de données, présentés dans cette section, sont utilisés dans les chapitres 4, 6, 7 et 8. Pour chacun d'entre eux, nous détaillons les processus de collecte et d'annotation, et nous expliquons le pré-traitement appliqué.

3.1.1 Jeu de données Waseem

Processus de collecte

Le jeu de données, nommé Waseem (Waseem and Hovy, 2016), a été créé à partir d'une collection de 136 052 tweets en langue anglaise. Les créateurs de Waseem se sont focalisés sur la détection de la haine envers les femmes (sexisme) et les immigrants (racisme). La collection s'est effectuée grâce à une application gratuite de Twitter permettant de récupérer des tweets publics. Afin d'affiner leurs recherches de tweets, les créateurs de Waseem ont utilisé des mots-clés pertinents, tels que *MKR*, *feminazi*, *immigrant*, *nigger*, etc., pour collecter le maximum de tweets haineux. Par exemple, ils ont analysé que le hashtag *#MKR*, qui fait référence à l'émission télévisée australienne *My Kitchen Rules*, fait l'objet de discours sexistes. Concernant la collecte des tweets racistes, les auteurs reportent l'utilisation du terme comme *muslims* qui est souvent utilisé dans les tweets racistes, mais aussi dans les tweets non-haineux. La collecte des tweets s'est effectuée sur 2 mois.

Processus d'annotation

Les auteurs ont procédé à une annotation manuelle des tweets sur une partie des tweets collectés, avec une vérification d'une annotatrice externe afin d'atténuer l'introduction de préjugés de la part des annotateurs. Le jeu de données Waseem contient 16 914 tweets annotés. L'accord inter-annotateurs obtenu est de 84%. Les annotateurs ont réparti les tweets en trois classes : *raciste*, *sexiste* et *neutre*. Le jeu de données final de Waseem est proposé sous la forme d'une liste d'identifiants de tweets avec l'annotation correspondante. Malheureusement, lors de la récupération des tweets grâce à leurs identifiants, nous n'avons pu télécharger que 10 900 tweets, les autres ayant été supprimés par la modération de Twitter. Les auteurs de Waseem ne proposent pas de partition en sous-ensemble d'apprentissage, de validation et de test de leur jeu de données.

3.1.2 Jeu de données SemEval 2019 tâche 5 : HatEval

Processus de collecte

Le jeu de données de la campagne d'évaluation de SemEval 2019 tâche 5 (Basile et al., 2019), nommé HatEval par la suite, est un jeu de données multi-langues de tweets haineux envers les femmes et les immigrants. Les tweets ont été collectés entre juillet et septembre 2018. Cette collecte s'est effectuée selon le processus suivant : identification de victimes potentielles, récupération des historiques des comptes Twitter postant des tweets haineux, utilisation de mots clés pour le filtrage des tweets. Le but des créateurs de HatEval était de proposer un jeu de données multi-langue pour la campagne d'évaluation SemEval 2019 : une partie en langue anglaise et une partie en langue espagnole. Il est important de noter que dans cette thèse nous utilisons uniquement la partie en langue anglaise. Nous notons qu'une partie des tweets haineux envers les femmes pour la langue anglaise ont été récupérée du jeu de donnée de (Caselli et al., 2018).

Processus d'annotation

Le jeu de données HatEval a été annoté par trois annotateurs, suivi de deux experts. Les tweets sont étiquetés en deux classes : *non-haine* et *haine*. Le jeu de données possède une annotation plus précise pour les tweets haineux : la cible du tweet (groupe ou individu) et l'agressivité du tweet (agressif ou non). Dans cette thèse, nous n'utilisons pas l'annotation de la cible du tweet et de l'agressivité de celui-ci. Nous notons que l'accord inter-annotateur de l'étiquetage des tweets est de 83%. Le jeu de données HatEval comprend 13 000 tweets, répartis en trois ensembles : entrainement, validation et test.

3.1.3 Jeu de données Founta

Processus de collecte

Les tweets qui ont permis la création du jeu de données Founta (Founta et al., 2018) ont été collectés aléatoirement (environ 32 millions) entre le 30 mars et le 9 avril 2017. Des filtres ont été appliqués afin d'écarter les spams, les tweets non anglais, etc. Le but des créateurs de Founta est d'annoter un grand jeu de données en termes de discours de haine.

Processus d'annotation

L'annotation du jeu de données Founta s'est effectuée en 3 étapes. Premièrement, environ 300 tweets ont été annotés manuellement avec une annotation participative (*crowdsourcing*). Ce

petit jeu de données a permis d'analyser les données pour l'annotation d'un plus grand jeu de données. Une sélection de 100 000 tweets a été effectuée avec une technique d'échantillonage aléatoire renforcée (*boosted random sampling* en anglais). L'annotation finale s'est aussi effectuée avec une annotation participative sur cette sélection de 100 000 tweets. Chaque tweet s'est vu annoté par 5 à 20 annotateurs différents. Les tweets sont classifiés en quatre classes : *normal*, *abusif*, *haine* et *spam*. Il est important de noter que dans cette thèse nous n'utilisons pas les tweets étiquetés comme *spam*, car nous nous focalisons sur la détection des contenus haineux. Il n'existe pas de répartition en ensemble d'apprentissage, de validation et de test pour ce jeu de données.

3.1.4 Jeu de données Davidson

Processus de collecte

Le jeu de données Davidson (Davidson et al., 2017) provient d'une collection de 85,4 millions de tweets postés par 33 458 comptes d'utilisateur de Twitter. Un lexique de termes à caractère offensant et haineux a été utilisé afin d'identifier les comptes des utilisateurs de Twitter postant des tweets potentiellement haineux. Ce lexique provient de `Hatebase.org` qui contient une grande base de données de termes à caractère offensant ou haineux. Une sélection d'environ 25 000 tweets parmi les millions de tweets collectés ont été choisis aléatoirement pour l'annotation du jeu de données final.

Processus d'annotation

Les créateurs du jeu de données Davidson ont fait le choix d'une annotation participative. Les annotateurs ont eu pour objectif d'annoter les tweets en trois classes : *haine*, *offensif* et ni l'un ni l'autre (classe *normal*). Le choix de séparer les tweets offensifs des tweets haineux est dû au fait qu'un tweet offensif n'est pas obligatoirement haineux. Par exemple, dire "*Ferme-la, tu n'as rien à dire !*" est offensif, mais pas haineux alors que "*Ferme-la, tu n'as rien à dire, tu es un immigré !*" est haineux. La version finale du jeu de données Davidson contient 24 800 tweets, chacun annoté par un minimum de trois annotateurs. L'accord inter-annotateur obtenu à la fin de l'annotation participative s'élève à 92%. Les auteurs de ce jeu de données ne proposent pas de répartition en sous-ensemble d'entraînement, de validation et de test.

3.1.5 Pré-traitements des données

Nous effectuons les pré-traitements identiques sur les quatre jeux de données présentés ci-dessus. Nous supprimons les hashtags (ex : *#BuildThatWall*), les mentions d'utilisateurs (ex : *@ceramisch*), le symbole des retweets (ex : *RT*), les adresses de messagerie électronique (ex : prenom.nom@domaine.fr) et les URL (ex : https://t.co/f66e68hLQq). Nous remplaçons les émojis (ou unicodes d'émojis) par leurs interprétations textuelles. Par exemple, l'émoji ❤ devient *:red_heart:* dans le tweet. Après l'application du pré-traitement, nous supprimons les tweets ne contenant aucun caractère des ensembles d'entraînement et de validation. Pour l'ensemble de test, nous étiquetons les tweets ne contenant aucun caractère comme *normal*. La table 3.1 contient des exemples du pré-traitement effectué.

Pour une meilleure lecture et compréhension des exemples dans la suite de la thèse, nous présenterons les exemples contenant des émojis sous leurs formes normales (😍🙈), et non sous leurs formes pré-traitées (*:smiling_face_with_heart_eyes: :see_no_evil_monkey:*).

Avant pré-traitement	*What a glorious cunt you have. Too bad youre a basic bitch. Always were always will be. Stay mad. Hoe. #Imtalkingaboutadude* 😂 😂
Aprés pré-traitement	*What a glorious cunt you have. Too bad youre a basic bitch. Always were always will be. Stay mad. Hoe. :face_ with_ tears_ of_ joy: :face_ with_ tears_ of_ joy:*
Avant pré-traitement	*@AnimeFemsplainr All women deserve an opinion and have the right to, you on the other hand my dear are a cunt who d... https://t.co/f66e68hLQq*
Après pré-traitement	*All women deserve an opinion and have the right to, you on the other hand my dear are a cunt who d...*
Avant pré-traitement	*RT @_ BPalacios : Bro 13 reasons is fucking me up rn 😭*[11]
Après pré-traitement	*Bro 13 reasons is fucking me up rn :loudly_ crying_ face:*
Avant pré-traitement	*@ScrewedHumans @Debunk_ The_ Left @DiamondandSilk @realDonaldTrump #BuildTheWallNow #BuildThatWall #TrumpTrain2020 @tracyf822 @mike_ bruiser347 @Edwardfart2018 @WhatevNvm @Gadsden_ Flagger @LaunaSallai @ddumas29 @wsj7707 @zebrark @Nudist4trump @chrissussdorf*
Après pré-traitement	

TABLE 3.1 – Exemples de tweets avant et après l'application des pré-traitements. Les tweets sont extrais des différents jeux de données présentés précédemment.

3.1.6 Statistiques des jeux de données

Le jeu de données Waseem comprend 27% de tweets sexistes ou racistes. Nous avons décidé de réunir les étiquettes *sexiste* et *raciste* sous une même étiquette "haine", car nous nous focalisons sur la détection de la parole haineuse de manière générale. Concernant le jeu de données Founta, nous avons décidé d'écarter les tweets classés *spam* puisque nous nous focalisons sur la détection des discours de haine. La table 3.2 renseigne les statistiques sur les différents jeux de données pour la détection de la parole haineuse. Comme le montre ce tableau, seul le jeu de données HatEval possède une distribution de tweets équilibrée entre les classes *normal* et *haine*. Les autres jeux de données ont une distribution des classes non-balancée et sont composés de très peu de tweets haineux comme Davidson et Founta qui comptent seulement 7% et 6% des tweets annotés haineux.

Jeux de données	**Waseem**	**HatEval**	**Davidson**	**Founta**
#Tweets	10 900	13 000	24 800	85 960
Normal	73%	58%	16%	63%
Offensif/Abusif	-	-	77%	31%
Haine	27%	42%	7%	6%

TABLE 3.2 – Nombre et distribution des tweets dans les jeux de données Waseem, HatEval, Davidson et Founta.

Afin de mener à bien nos expériences, nous séparons les jeu de données de Waseem, Davidson et Founta, en trois sous-ensembles : entraînement, validation et test. La table 3.3 montre le nombre de tweets et la distribution des différents sous-ensembles de données pour chacun des

11. Code HTML pour l'émoji : 😭

jeux de données présenté. Pour le jeu de données Waseem, nous avons décidé de séparer les jeu de données avec la distribution suivante : 80% pour l'ensemble d'entraînement, 10% pour l'ensemble de validation et 10% pour l'ensemble de test. Nous appliquons cette séparation car cette distribution est souvent utilisée sur ce jeu de données (Mozafari et al., 2020; Bose, 2023). Concernant HatEval, nous utilisons la distribution par défaut, utilisée lors de la campagne d'évaluation SemEval 2019 task 5. Pour les jeux de données Founta et Davidson, étant les deux plus grand jeux de données (en termes de nombre de tweets), nous effectuons la distribution suivante : 60% de tweets d'entraînement, 20% de validation et 20% de test. Les différents sous-ensembles des jeux de données ont été créés aléatoirement en gardant la distribution des classes, avec la librairie python *Scikit-learn*[12].

Jeux de données	**Waseem**	**HatEval**	**Davidson**	**Founta**
Apprentissage (%)	8640 (80%)	8969 (60%)	14866 (60%)	51491 (60%)
Validation (%)	1077 (10%)	997 (10%)	4955 (20%)	17158 (20%)
Test (%)	1090 (10%)	3000 (30%)	4956 (20%)	17194 (20%)

TABLE 3.3 – Nombre de tweets avec le pourcentage de répartitions pour chacun des sous-ensembles d'apprentissage, de validation et de test, pour les différents jeux de données après les pré-traitements.

3.1.7 Mesure d'évaluation

Pour évaluer nos travaux expérimentaux, nous utilisons comme mesure d'évaluation le score *macro-F1*, qui correspond à la moyenne des scores F1 obtenus pour chaque classe (C signifie le nombre de classes dans l'équation ci-dessous), selon la définition suivante :

$$\textbf{\textit{macro-F1}} = \frac{1}{C}\sum_{i=1}^{C} F1_i \qquad \text{avec } F1_i = 2 \times \frac{(\text{précision}_i \times \text{rappel}_i)}{(\text{précision}_i + \text{rappel}_i)}$$

$$\text{précision} = \frac{\text{nombre de vrais positifs}}{\text{nombre d'instances prédites}} \qquad \text{et} \quad \text{rappel} = \frac{\text{nombre de vrais positifs}}{\text{nombre d'instances positives}}$$

Cette mesure permet de prendre en compte la distribution déséquilibrée des classes présente dans les différents jeux de données. De plus, afin de déterminer s'il y a une amélioration significative entre des systèmes de base et nos approches, nous utilisons un test de paires appariées avec un risque de 5% (Gillick and Cox, 1989).

3.2 Les jeux de données pour l'identification des expressions polylexicales

Dans cette section nous présentons les jeux de données utilisés pour l'identification automatique des EP dans les tweets (utilisés dans le chapitre 5). Nous utilisons des jeux de données annotés en termes d'EP disponibles dans la littérature : DiMSUM (Schneider et al., 2016), Streusle (Schneider and Smith, 2015) et PARSEME (Ramisch et al., 2018).

12. `https://scikit-learn.org`

3.2.1 Streusle

Streusle est un jeu de données contenant des critiques en ligne provenant du jeux de données English Web Treebank (Bies et al., 2012) qui sont annotées en termes de super-sens [13] et d'EP *faibles* (par exemple, *narrow escape*, *do not be surprised*) et *fortes* (par exemple, *go out of my way*, *close call*) (Schneider and Smith, 2015). Une EP est considérée comme faible si une formulation alternative est acceptable. Par exemple, l'unité lexicale (UL) *narrow* de l'EP *narrow escape* (*échapper de justesse* en français) pourrait être remplacer par l'UL *close* qui donnera *close escape* qui a la même signification que *narrow escape*. A l'inverse, une EP est considérée comme forte si elle est complètement opaque, c'est à dire qu'aucun des mots de l'EP ne peut être substitués tout en exprimant le même sens. Par exemple, l'EP *close call* (*éviter de justesse* en français) est considérée comme une EP forte.

Les textes contenus dans ce jeu de données sont en anglais. Le jeu de données est annoté manuellement en 20 catégories d'EP (présentées dans le chapitre 5). Nous utilisons la version 4.3 du jeu de données Streusle [14]. L'annotation des EP est au format "BIO" (*Begin-Inside-Outside*) avec la concaténation des catégories d'EP et du super-sens des mots. La table 3.4 illustre un exemple des étiquettes d'EP utilisées dans le jeu de données Streusle. L'étiquette du premier mot composant une EP est "B" (*begin*), les mots suivants de l'EP sont étiquetés "I" (*inside*), et les mots n'appartenant à aucune EP sont étiquetés "O". Il concatène aux étiquettes "B" les catégories des EP et les étiquettes de super-sens, aux étiquettes "I" un marqueur "_", et aux étiquettes "O" le super-sens. Nous notons qu'il utilise également des étiquettes supplémentaires "b", "i" et "o" ayant la même signification que les étiquettes "B", "I" et "O", mais indiquant que les UL sont imbriquées dans une EP.

Le jeu de données est divisé en trois sous-ensembles : apprentissage, validation et test.

3.2.2 PARSEME

Le jeu de données **PARSEME** (Ramisch et al., 2018) contient des news, des extraits de site internet et de Wikipédia. Ce jeu de données a été développé pour la campagne d'évaluation PARSEME, qui proposait des jeux de données annotés en termes d'EP dans plus de 18 langues. PARSEME est annoté uniquement en termes d' EP verbales fortes. Six catégories d'EP verbales sont considérées. Le jeu de données est au format *cupt* [15] et utilise le format d'étiquette propre à PARSEME. La table 3.4 illustre un exemple des étiquettes d'EP au format PARSEME. Le mot commençant une EP aura l'étiquette "*i:cat*", où "*i*" est l'identifiant de l'EP et "*cat*" est la catégorie de l'EP et les mots appartenant à cette EP seront étiquetés par l'identifiant "*i*. Les mots n'appartenant pas à une EP seront étiquetés "*". Le jeu de données est divisé en ensemble d'apprentissage et de test, sans ensemble de validation.

3.2.3 DiMSUM

Le jeu de données **DiMSUM** (Schneider et al., 2016) contient des critiques en ligne provenant du jeux de données English Web Treebank, des transcriptions de discussions (TED talk) et des tweets. Il a été développé dans le cadre de la campagne d'évaluation SemEval 2016 tâche 10. Nous utilisons uniquement la partie tweet car nous concentrons nos expériences sur les tweets, et qu'une grande partie des données non-tweets sont incluses dans le jeux de données Streusle. Le jeu de données est annoté manuellement en termes d'EP fortes sans les étiquettes de catégories

13. Classe sémantique d'un mot.
14. `https://github.com/nert-nlp/streusle`
15. Format en colonne ConLL avec une colonne supplémentaire pour l'annotation des EP.

d'EP correspondantes. Ce jeu contient toutes les catégories d'EP comme Streusle, cependant ces catégories ne sont pas mentionnées dans l'étiquetage des données. Le jeu de données est au format *ConLL* (format en colonne) et utilise un format d'étiquette "BIO" pour l'annotation des EP comme le jeu de données de Streusle. Nous notons que les étiquettes ne contiennent ni les catégories des EP, ni l'annotation en termes de super-sens. Néanmoins, il contient l'étiquetage pour signifier l'imbrication d'UL entre des mots d'une EP : "b", "i" et "o" qui ont la même signification que les étiquetes "B", "I" et "O. La table 3.4 illustre un exemple de l'annotation des EP dans le jeu de données DiMSUM. Le jeu de données est divisé en ensemble d'apprentissage et de test.

Mots	**Streusle**	**PARSEME**	**DiMSUM**
I	O-PRON	*	O
had	B-V.LVC.full-v.social	1:V.LVC.full	B
a	o-DET	*	o
routine	o-ADJ	*	o
surgery	I_	1	I
for	O-P-p.Purpose	*	O
an	O-DET	*	O
ingrown	B-N-n.BODY	2:N	B
toenail	I_	2	I
.	O-PUNCT	*	O

TABLE 3.4 – Représentation des différents formats d'étiquettes d'EP pour l'exemple :"*I had a routine surgery for an ingrown toenail.*". La phrase contient deux EP : *have surgery* et *ingrown toenail*.

3.2.4 Pré-traitements des données

Les trois jeux de données, présentés ci-dessus, sont tous proposés dans un format différent. Nous avons décidé de les mettre sous le même format et nous avons choisi d'utiliser le format *cupt* de PARSEME pour lequel il existe des boîtes à outils afin de manipuler aisément ce type de format. Comme les jeux de données de PARSEME et de DiMSUM sont annotés en termes d'EP fortes, nous ne prenons en compte que les annotations d'EP fortes du jeu de données de Streusle, les EP faibles ne sont pas prises en compte sauf pour une configuration d'apprentissage particulière (voir section 5.5). Concernant le pré-traitement de DiMSUM, certains tweets sont normalisés, alors nous normalisons tous les autres tweets de la même manière : nous remplaçons les mentions d'utilisateur par un token unique (ex : *@toto–>@USER*). Nous faisons de même concernant les URL (ex : *https://t.co/f66e68hLQq –> URL*).

Pour tous les jeux de données, à l'exception de DiMSUM test, nous filtrons les EP : dans une phrase donnée, lorsqu'un mot est commun à deux EP (chevauchement d'EP) ou que deux EP sont imbriquées, on supprime la deuxième EP ou l'EP la plus courte (en nombre de mots) si elles commencent à la même position dans la phrase. La table 3.5 illustre le filtrage effectué pour les chevauchements et les imbrications d'EP. Ces phénomènes sont peu fréquents et se produisent dans moins de 5% des phrases. De plus, la modélisation du chevauchement des EP pour l'apprentissage d'une système d'identification d'EP est complexe, car il faudrait soit dupliquer les phrases avec les différentes EP, soit utiliser des schémas d'étiquetages complexes concaténant les différentes EP. Cela pourrait biaiser l'apprentissage de systèmes automatiques. Concernant l'imbrication des EP, il serait difficile pour un système automatique d'apprendre à généraliser sur les EP imbriquées car elles sont très peu présentes dans les jeux de données. Après l'application

du filtrage, nous obtenons les statistiques présentées dans la table 3.6.

Exemple d'imbrication d'EP			**Exemple de chevauchement d'EP**		
Mots	**Avant filtrage**	**Après filtrage**	**Mots**	**Avant filtrage**	**Après filtrage**
I	*	*	He	*	*
have	1:V.LVC.full	1:V.LVC.full	made	1:V.LVC.full,2:V.LVC.full	1:V.LVC.full
a	2:N	*	a	*	*
bit	2	*	presentation	1	1
of	*	*	and	*	*
experience	1	1	a	*	*
watching	*	*	tribute	2	*
the	*	*	at	*	*
usual	*	*	the	*	*
assembly	3:N	2:N	same	*	*
line	3	2	time	*	*
.	*	*	.	*	*

TABLE 3.5 – Exemples du filtrage utilisé supprimant les EP imbriquées et les chevauchements d'EP. L'exemple "*I have a bit of experience watching the usual assembly line.*" possède une imbrication d'EP avec les EP*have experience* et *a bit*. Le filtrage supprimera l'EP *a bit*. L'exemple "*He made a presentation and a tribute at the same time.*" possède un chevauchement d'EP avec les EP *make presentation* et *make tribute*. Le filtrage gardera l'EP *make presentation*.

3.2.5 Statistiques des jeux de données

Jeux de données		**#phrases**	***#Unités lexicales***	**#EP**
Streusle	App.	2 724	44 822	2 425
	Val.	554	5 394	283
	Test	535	5 381	281
PARSEME	App.	3 471	53 201	331
	Test	3 965	71 002	501
DiMSUM partie tweet	App.	987	18 247	1 112
	Test	500	6 627	362

TABLE 3.6 – Nombre de phrases, d'unités lexicales et d'occurrences d'EP des partitions standards dans les ensembles d'apprentissage (*App.*), de validation (*Val.*) et de test pour les jeux de données Streusle, PARSEME et DiMSUM.

La table 3.6 montre la distribution des différents jeux de données introduit ci-dessous. Le jeu de données Streusle contient 3 813 phrases qui ne sont pas des tweets avec environ 3 000 EP fortes annotées manuellement. Les ensembles d'apprentissage, de validation et de test de Streusle contiennent respectivement 2 425, 283 et 281 EP annotées. Le jeu de données PARSEME contient 7 436 phrases pour seulement 832 EP. PARSEME est le jeu de données contenant le plus de phrases, mais aussi le moins d'EP annotées étant donnée qu'il contient uniquement des EP verbales. Les ensembles d'apprentissage et de test contiennent respectivement 331 et 501 EP verbales annotées. Le jeu de données DiMSUM est le plus petit contenant moins de 1500 phrases. Cependant, c'est le seul jeu de données qui contient des tweets. DiMSUM possède plus de 1 112

EP annotées dans l'ensemble d'apprentissage et 362 EP dans l'ensemble de test (qui nous servira d'ensemble de test pour nos évaluations expérimentales).

3.2.6 Mesures d'évaluation

Dans cette section, nous décrivons les mesures d'évaluation utilisées pour étudier la robustesse des systèmes d'identification d'EP. Nous utilisons des mesures standards qui ont été appliquées lors des campagnes d'évaluation PARSEME (Savary et al., 2017) et DiMSUM (Schneider et al., 2016). Nous utilisons l'exemple de la table 3.7 pour expliquer les mesures d'évaluation.

Toutes les mesures décrites ci-dessous sont basées sur le score F1, qui est une moyenne harmonique de la précision et du rappel. Le score F1 prend en compte ces deux mesures, où la précision mesure la proportion de vrais positifs parmi les instances prédites et le rappel mesure la proportion de vrais positifs parmi toutes les instances annotées. Le score F1 est donné par la formule suivante :

$$F1 = 2 \times \frac{(\text{précision} \times \text{rappel})}{(\text{précision} + \text{rappel})}$$

$$\text{avec précision} = \frac{\text{nombre de vrais positifs}}{\text{nombre d'instances prédites}} \quad \text{et} \quad \text{rappel} = \frac{\text{nombre de vrais positifs}}{\text{nombre d'instances positives}}$$

La mesure ***MWE-based*** correspond au score F1 des EP entièrement prédites. Une EP est considérée comme vraie positive si toutes les UL qui la composent sont correctement prédites. Si l'EP est partiellement étiquetée, elle est considérée comme fausse. En prenant l'exemple de la table 3.7, le score *MWE-based* est de 50% car une EP sur les deux (l'EP ***take care of***) est correctement identifiée.

La mesure ***Token-based*** correspond au score F1 des UL appartenant à une EP et évalue les correspondances partielles. Contrairement à la mesure *MWE-based*, cette mesure s'applique au niveau des UL. Pour l'exemple de la table 3.7, la précision est de 6/6 (100%) et le rappel est de 6/7 (85,7%). Ce qui nous donne une *Token-based* de 92,3% en appliquant la formule du score F1 présentée ci-dessus.

La mesure ***MWE-link-based*** correspond au score F1 basé sur la correspondance des paires de mots adjacents et prend en compte les EP partiellement correctes. Cette mesure est une variante de la mesure *token-based* en se concentrant sur des bi-grammes d'UL. En se basant sur l'exemple de la table 3.7, la prédiction contient 4 paires de mots adjacents prédites (*went-out*, *out-way*, *take-care*, *care-of*) et les paires de mots adjacents attendus sont au nombre de 5 (*went-out*, *out-of*, *of-way*, *take-care*, *care-of*). Donc nous avons 3 paires de mots adjacents vrais positifs (*went-out*, *take-care*, *care-of*) et nous obtenons une précision de 3/4 (75%) et le rappel de 3/5 (60%). Nous obtenons finalement une *MWE-link-based* de 67%.

En plus de ces mesures standards, nous évaluons les systèmes d'identification des EP plus finement en utilisant trois mesures d'évaluation basées uniquement sur le score F1 pour les EP entièrement prédites (*MWE-based*).

La mesure ***F-non-vue*** fait référence aux EP correctement détectées dont les formes canoniques des UL ne sont pas observées dans l'ensemble l'apprentissage. Cette mesure permet d'observer si le système réussit à généraliser à partir de l'ensemble d'apprentissage.

La mesure ***F-variante*** reflète les EP correctement prédites dont les formes canoniques des UL apparaissent dans l'ensemble d'apprentissage, mais leurs formes textuelles sont différentes : par exemple, ***have surgery*** dans l'ensemble d'apprentissage et ***had surgery*** dans l'ensemble de test. Cette mesure permet d'observer la robustesse du système par rapport à la variabilité des EP.

La mesure ***F-identique*** fait référence aux EP correctement détectées dont les formes textuelles sont observées lors de l'apprentissage.

Phrase	They	**went**	**out**	**of**	their	**way**	to	**take**	**care**	**of**	my	cat	.
Attendu		1	1	1		1		2	2	2			
Prédiction		1	1			1		2	2	2			

TABLE 3.7 – Exemple pour illustrer les différentes mesures d'évaluation. L'exemple contient deux EP (en gras) : ***went out of way*** et ***take care of***.

3.3 Résumé du chapitre

Dans ce chapitre, nous avons présenté les jeux de données utilisés pour la détection de la parole haineuse (Section 3.1), ainsi que pour l'identification des EP dans les tweets (Section 3.2). Pour la tâche de la détection automatique de la parole haineuse, nous basons nos expériences sur quatre jeux de données de tweets : Waseem, HatEval, Davidson et Founta (Waseem and Hovy, 2016; Basile et al., 2019; Davidson et al., 2017; Founta et al., 2018). Les tweets des jeux de données Waseem et HatEval sont catégorisés en deux classes : *haineux* et *non-haineux*. Concernant Founta et Davidson, les tweets sont divisés en trois classes : *haineux*, *offensif* ou *abusif* et *normal*. Pour rappel, les jeux de données annotés en termes de haine sont utilisés dans les chapitres 4, 6, 7 et 8.

Concernant la tâche d'identification des EP dans les tweets, nous utilisons trois jeux de données contenant des critiques en ligne, des news, des extraits d'internet et de Wikipedia, et des tweets. Les trois jeux de données se nomment DiMSUM, Streusle et PARSEME (Schneider et al., 2016; Schneider and Smith, 2015; Ramisch et al., 2018). Les jeux de données pour l'identification des EP dans les tweets sont utilisés dans le chapitre 5.

Pour chacune des deux tâches, nous avons exposé les pré-traitements appliqués aux différents jeux de données. Nous avons également présenté, pour chacune des tâches traitées dans cette thèse, les mesures d'évaluation utilisées pour quantifier nos résultats expérimentaux.

4

Intégration de caractéristiques linguistiques dans des réseaux neuronaux

Dans ce chapitre, nous présentons notre étude sur l'intégration de caractéristiques linguistiques dans un système neuronal. Le but de ce chapitre est d'établir quelles caractéristiques sont utiles pour la détection des discours haineux.

Nous proposons d'étudier les caractéristiques suivantes : les mots contenus dans un dictionnaire de termes haineux, la casse des mots, la répétition de la ponctuation, les émojis et la partie du discours (POS pour *part of speech* en anglais). Nous proposons d'intégrer ces caractéristiques dans un système neuronal fondé sur des plongements (*embeddings* en anglais) de phrases. Ici, nous étudions différentes caractéristiques qui ont déjà été utilisées pour la détection des discours de haine, mais jamais en tant que caractéristiques additionnelles aux plongements de phrases.

Ce chapitre s'articule de la façon suivante : nous présentons notre méthodologie dans la section 4.2. Ensuite, nous décrivons les configurations expérimentales dans la section 4.3. Nous exposons les résultats obtenus dans la section 4.4.

4.1 Motivations

Comme mentionné dans le chapitre 2 beaucoup de caractéristiques différentes ont été étudié pour la détection des discours de haine (Chen et al., 2012; Gitari et al., 2015; Nobata et al., 2016b; Burnap and Williams, 2015; Davidson et al., 2017; Corazza et al., 2020; Gambino and Pirrone, 2020). Par exemple, Nobata et al. (2016a) ont montré que la combinaison de caractéristiques linguistiques, telles que les mots haineux, les POS ou bien la casse des mots (par exemple, le nombre de caractères en majuscule dans un mot), avec des plongements de mots en entrée d'un modèle de décision obtient de meilleurs résultats dans la détection des discours haineux comparé à l'utilisation indépendante de ces caractéristiques. De nombreux travaux ont montré l'utilité des termes injurieux et haineux pour détecter les discours haineux (Chen et al., 2012; Gitari et al., 2015; Davidson et al., 2017). Corazza et al. (2020) ont montré que l'utilisation des émojis aide à la détection des discours haineux des tweets en anglais avec un système neuronal basé sur des plongements de mots. Les auteurs ont proposé deux méthodes pour représenter les émojis : sous forme textuelle ou avec des plongements d'émojis. Gambino and Pirrone (2020) ont montré que les POS peuvent être utile pour la détection des discours haineux lors de la campagne d'évaluation EVALITA (Basile et al., 2020). Des plongements de mots ont aussi été étudiés pour notre tâche et

ils se sont montrés être plus efficace que les n-grammes de mots ou de caractères (Badjatiya et al., 2017; Gambäck and Sikdar, 2017b; Zhang et al., 2018). Néanmoins, les plongements de phrases semblent être de meilleures caractéristiques que les plongements de mots pour la détection des discours de haine (Djuric et al., 2015; Indurthi et al., 2019). Indurthi et al. (2019) ont montré que l'utilisation des plongements de phrases pour représenter des tweets obtenait de bons résultats dans la détection des discours haineux, comme en témoigne la première place de leur système lors de la campagne d'évaluation SemEval 2019, tâche 5 (Basile et al., 2019).

Dans ce chapitre nous nous intéressons à l'ajout de caractéristiques linguistiques (les termes haineux, la casse des mots, les POS, les émojis et la ponctuation) dans un système neuronal fondé sur les plongements de phrases. L'idée est de répondre à la question suivante :

— Quelles caractéristiques linguistiques, en complément des plongements de phrases, sont pertinentes pour la détection des discours de haine ?

4.2 Méthodologie proposée

Dans cette section, nous établissons notre méthodologie en décrivant le système de base utilisé pour ajouter les caractéristiques linguistiques étudiées. Ensuite, nous présentons l'architecture de notre système et les différentes caractéristiques linguistiques utilisées.

4.2.1 Architecture du système de base

Nous utilisons un système de base fondé sur les plongements de phrases pour comparer expérimentalement notre approche. Comme mentionné précédemment, Indurthi et al. (2019) ont démontré que des plongements de phrases sont efficaces pour la détection des discours de haine. De plus ce système a obtenu la première place lors de la campagne d'évaluation SemEval 2019 tâche 5 (Basile et al., 2019) surpassant l'affinage des poids (*fine-tuning* en anglais) du modèle de langage BERT (Devlin et al., 2019). Nous rappelons que dans le système FERMI les plongements de phrases sont donnés en entrée d'un classifieur traditionnel de type machine à vecteur de support (SVM pour *Support Vector Machine*). Dans notre système de base, nous utilisons des couches Dense pour apprendre à classifier les tweets à la place de l'utilisation d'un classifieur SVM. En effet, des chercheurs ont montré que les réseaux neuronaux sont plus performants que les classifieurs traditionnels pour la détection des discours de haine (Badjatiya et al., 2017). La figure 4.1 présente le système de base. En entrée nous donnons les plongements de phrases représentant les tweets qui passe par deux couches de projection (*Dense*).

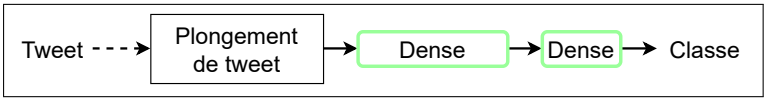


FIGURE 4.1 – Système de base inspiré de (Indurthi et al., 2019) utilisant les plongements de phrases en entrée dee couches neuronales pour la classification des tweets.

4.2.2 Architecture du système proposé

Pour ajouter les caractéristiques linguistiques en supplément des plongements de phrases dans un réseau neuronal, plusieurs possibilités existes : intégrer les caractéristiques au niveau des mots ou des phrases. Une solution simple est de les intégrer au niveau de la phrase en concaténant les caractéristiques linguistiques aux plongements de phrases. Une autre solution est d'intégrer les

caractéristiques linguistiques au niveau des mots. Nous avons opté pour cette deuxième solution. Les caractéristiques linguistiques risques d'être noyées dans un vecteur de grandes dimensions issu de la concaténation de celles-ci avec des plongements de phrases et donc ne pas être ignorées par un système neuronal.

Nous avons développé une architecture neuronale basée sur deux branches : la première pour les plongements des phrases qui sont des caractéristiques au niveau de la phrase et la seconde pour les caractéristiques linguistiques supplémentaires qui sont au niveau des mots. La figure 4.2 montre l'architecture de notre système. Pour la première branche neuronale, comme entrée nous utilisons des plongements de phrases pour représenter les tweets (voir le bloc *plongement de tweet* dans la figure 4.2). Une projection (couche Dense) prend en entrée ces plongements de phrases avant d'être concaténée avec la sortie de la deuxième branche neuronale.

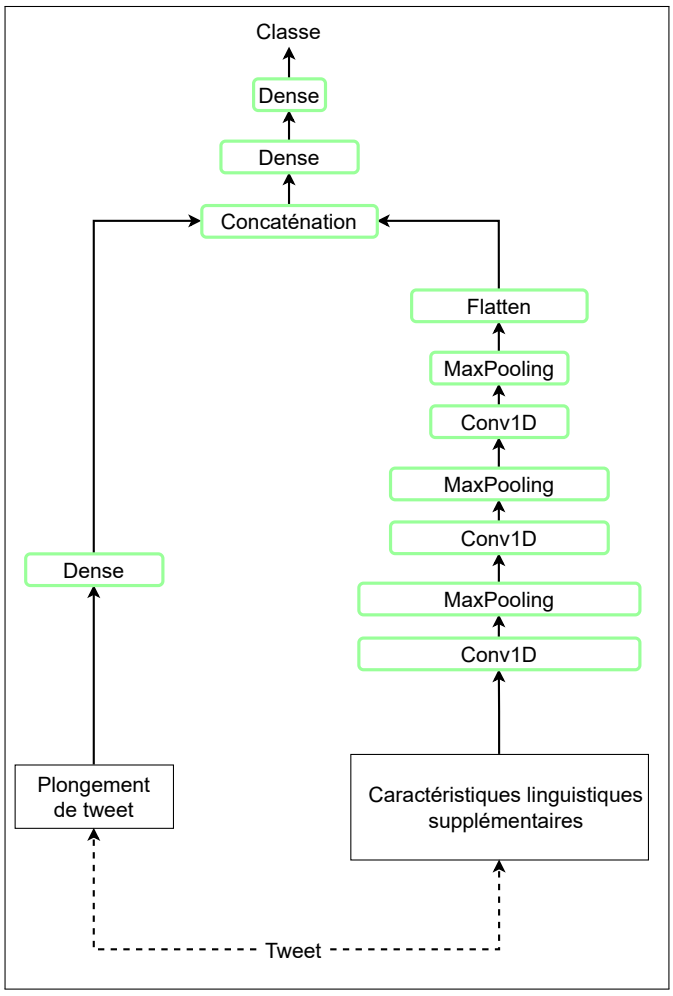


FIGURE 4.2 – Proposition d'un système neuronal basé sur les plongements de phrases et intégrant des caractéristiques linguistiques.

Concernant la deuxième branche neuronale, en entrée nous représentons les caractéristiques linguistiques qui passe par trois couches neuronales convolutionnelles à une dimension (Conv1D) avec une réduction de dimension (*maxpooling*), dont la sortie de la dernière couche de réduction de dimension est transformée en un vecteur à une dimension (*flatten*). Nous avons également testé d'autres types de couches neuronales, tels que les couches neuronales récurrentes (par exemple,

des couches LSTM), à la place des blocs de couches convolutionnelles, mais nous n'avons pas constaté d'amélioration. Les sorties des deux branches neuronales sont concaténées et utilisées comme entrée pour deux couches denses : la première couche dense est utilisée pour réduire la dimension de la concaténation des deux branches, tandis que la deuxième couche dense permet d'obtenir la réponse de la classification.

4.2.3 Caractéristiques linguistiques utilisées

Ici, nous décrivons les caractéristiques linguistiques utilisées dans notre système en complément des plongements de phrases.

Les mots à caractère haineux (MH) nous semble une caractéristique importante à prendre en compte par un système automatique pour détecter la haine. Cette caractéristique fait partie des caractéristiques utilisées pour la détection des discours haineux (Njagi et al., 2015; Nobata et al., 2016a). Ainsi, nous étiquetons automatiquement les mots haineux en utilisant un dictionnaire de termes haineux et les représentons au niveau des mots, comme illustré dans la table 4.1 par un vecteur unidimensionnel contenant une information binaire : "1" si le mot apparaît dans le dictionnaire, "0" sinon.

Tweet	**Fuck**	that	**hoe**	?	?	?	that	s	exactly	what	dese	**hoes**	doin	.
MH	1	0	1	0	0	0	0	0	0	0	0	1	0	0

TABLE 4.1 – Exemple de représentation de la caractéristique *MH* pour l'exemple "*Fuck that hoe ? ? ? that s exactly what dese hoes doin.*". Les mots haineux sont en gras dans l'exemple.

Les caractères de ponctuation (Punct) nous paraît une caractéristique intéressante pour aider un système à détecter les discours haineux. En effet, nous pensons que la répétition d'une ponctuation, comme *!!!* ou ..., peut indiquer une émotion contenue dans une phrase, comme dans les exemples "*California was attracting all kind of Iraqi refugees and migrants ....*" (tweet normal) et *Refugees need to go Home!!!* (tweet haineux). Nous représentons cette caractéristique au niveau des mots (comme dans la table 4.2), c'est à dire nous utilisons un vecteur unidimensionnel contenant des "1" si le symbole est une ponctuation, "0" sinon.

Tweet	Refugees	need	to	go	Home	!	!	!
Punct	0	0	0	0	0	1	1	1

TABLE 4.2 – Exemple de représentation de la caractéristique *Punct* pour l'exemple "*Refugees need to go Home!!!*".

La casse des mots (CM) nous semble une caractéristique pertinante également. En effet, nous émettons l'hypothèse que les mots contenant des majuscules traduisent une amplification du sens des mots, comme dans les exemples "*50k refugees would be the lowest refugee goal EVER. PLEASE help*" (tweet normal) et "*They want to close ALL women is prisons.*" (tweet haineux). Nous représentons cette caractéristique au niveau des mots (exemple dans la table 4.3). Nous étiquetons automatiquement les mots en utilisant les indications ci-dessous :

(1) le mot ne contient pas de majuscule ;

(2) seulement la première lettre du mot est en majuscule ;

(3) le mot contient au plus (inférieur ou égal) la moitié des caractères en majuscule, et ne rempli pas la condition (2).

(4) le mot contient au moins (strictement supérieur) la moitié des caractères en majuscule, et ne rempli pas la condition (5).

(5) tous les caractères du mot sont en majuscule.

L'étiquette (2) a pour objectif de représenter les noms propres, bien que cela ne soit pas toujours le cas dans les tweets. Les étiquettes (3) et (4) sont utilisées pour représenter une nuance de l'amplification. Nous faisons ce choix car l'utilisation du nombre de caractères en majuscules dans les mots pourrait biaiser le système lors de l'apprentissage. Cela permet d'homogénéiser les étiquettes pour cette caractéristique pour représenter cette caractéristique.

Tweet	NOBODY	cleans	a	house	FASTER	than	a	Nigga	expecting	some	PUSSyy	.
CM	5	1	1	1	5	1	1	2	1	1	4	1

TABLE 4.3 – Exemple de représentation de la caractéristique *CM* pour l'exemple "*NOBODY cleans a house FASTER than a nigga expecting some PUSSyy.*".

Les émojis (EmoVec) ont une signification linguistique et peuvent véhiculer des émotions. Pour représenter les émojis dans notre système nous utilisons la boîte à outils *emoji2vec* (Eisner et al., 2016), qui génère des plongements pour les émojis présents dans les tweets. Nous représentons également les émojis sous leur forme textuelle avec l'utilisation du pré-traitement introduit dans le chapitre 3.1.5.

Les parties du discours (POS) représentent une caractéristique syntaxique au niveau des mots. Étant donné que les tweets dans nos jeux de données ne sont pas annotés en termes de POS, nous les annotons automatiquement en termes de POS (au format *Universal Dependencies*[16]) avec la boîte à outils *twpipe* (Liu et al., 2018). Les POS sont réparties en 17 catégories, *DET* pour déterminant, *NOUN* pour nom, *VERB* pour verbe, etc. Pour cette caractéristique, nous représentons les mots par leurs POS, comme dans la table 4.4.

Tweet	These	niggas	ai	not	shit	but	hoes	with	tricks
POS	DET	NOUN	AUX	PART	VERB	CCONJ	VERB	ADP	NOUN

TABLE 4.4 – Exemple de représentation de la caractéristique *POS* pour l'exemple "*These niggas ai not shit but hoes with tricks*".

4.3 Configurations expérimentales

4.3.1 Génération des plongements de phrases

Le système de base ainsi que celui proposé sont fondés sur l'utilisation des plongements de phrases. Pour générer les plongements de phrases, nous utilisons l'encodeur *Universal Sentence Encoder* (Cer et al., 2018). Ces plongements de phrases se sont montré utiles pour la détection de la parole haineuse comme dans le système *FERMI* (Indurthi et al., 2019). Dans ce chapitre uniquement, nous utiliserons un pré-traitement différent de celui expliqué dans le chapitre 3.1.5 afin d'étudier l'efficacité des émojis sous formes textuelles pour notre tâche. Pour le système de base nous utilisons donc deux pré-traitements différents sur les tweets. Le premier pré-traitement est identique à celui introduit dans le chapitre 3.1.5 (nous nommons les plongements générés avec ce pré-traitement par *USE*). Le second est basé sur le même pré-traitement que dans le

16. `https://universaldependencies.org/`

chapitre 3.1.5 à l'exception que **les émojis (sous forme textuelle) sont supprimés**, afin de vérifier si les émojis sous forme textuelle apportent une information intéressante. Nous nommons les plongements de phrases générés avec ce pré-traitement par *USE** dans la suite du chapitre.

4.3.2 Paramètres des systèmes

Le système de base utilise deux couches dense successives : la première couche dense possède 256 neurones et la deuxième utilise 1 neurone avec une activation *sigmoid* pour Waseem et HatEval (car ils contiennent un étiquetage binaire) et 3 neurones avec une activation *softmax* pour Davidson et Founta (car ils possèdent trois classes : *normal*, *offensif* ou *abusif* et *haineux*).

Concernant notre proposition de système pour intégrer les caractéristiques linguistiques, les trois couches Conv1D successives utilisent 32, 16 et 8 filtres. Chacune des couches Conv1D possèdent une fenêtre de taille 3 et une activation d'unité linéaire rectifiée (ReLU pour *Rectified Linear Unit* en anglais) Ces paramètres sont issus d'une recherche des meilleurs paramètres sur l'ensemble de validation de HatEval. Également, nous avons testé différentes configurations à base de LSTM, mais nous n'avons pas remarqué d'amélioration significative. Après la concaténation des deux branches neuronales, les couches denses possèdent les mêmes configurations que les couches dense du système de base.

4.3.3 Paramètres d'apprentissage

Pour chacun des systèmes, nous effectuons cinq entraînements sur chaque ensemble d'apprentissage en utilisant une initialisation aléatoire. Chaque entraînement est limité à un maximum de 100 époques, avec l'utilisation d'un arrêt prématuré (*early stopping*) basé sur le taux d'erreur minimal atteint sur l'ensemble de validation, et une patience de cinq époques. Nous utilisons un optimiseur de type *Adam* avec un *learning rate* de 1e-3.

4.4 Résultats obtenus

Dans un premier temps, nous comparons l'utilisation des deux pré-traitements proposés. Dans un second temps, nous étudions notre proposition d'intégration de caractéristiques. Nous rappelons que les jeux de données relatifs à la haine et les mesures d'évaluation sont décrits dans le chapitre 3. La table 4.5 contient les scores médians de *macro-F1* sur les quatre ensembles de test utilisés.

Comparaison des deux pré-traitements : USE* versus USE

Ici, nous comparons les deux pré-traitements utilisés, avec les caractéristiques de *USE** (avec le pré-traitement supprimant les émojis) et de *USE* (utilisant le pré-traitement transformant les émojis dans leurs forme textuelle). Nous remarquons que la présence des émojis sous forme textuelle améliore, de manière significative étant donnée un test appariées avec un risque de 5% (Gillick and Cox, 1989), les résultats en moyenne sur l'ensemble des jeux de données : *USE* obtient 72,0% de score moyen de *macro-F1* comparé à 71,2% obtenu avec l'utilisation de *USE**. Nous constatons que les scores *macro-F1* sont plus élevés avec *USE* sur les ensembles de test de Waseem, Davidson et Founta, avec respectivement 79,5%, 72,2% et 72,4%, comparé à l'utilisation du pré-traitement sans les émojis *USE**. De plus, nous remarquons que sur les jeux de données Davidson et Founta, l'amélioration est significative avec l'utilisation du pré-traitement avec les émojis sous forme textuelle. Cependant, nous observons sur HatEval que *USE** obtient

un meilleur score *macro-F1* comparé à *USE*, sans différence significative entre les deux représentations des tweets : 64,3% avec *USE** et 63,9% avec *USE*. Nous en concluons que les émojis, représentés sous forme textuelle, aident à la détection des discours haineux.

Intégration des caractéristiques linguistiques

Maintenant, nous étudions les résultats de l'intégration des différentes caractéristiques linguistiques dans notre système. Nous constatons que l'utilisation des caractéristiques *CM*, *MH*, *Punct*, et *POS* obtiennent des scores moyens de *macro-F1* légèrement inférieurs comparés au système de base *USE* sur tous les ensembles de test. Le système de base avec *USE*, n'utilisant pas les caractéristiques linguistiques obtient 72,0% de score moyen de *macro-F1* comparé à 71,9%, 71,7%, 71,9% et 71,6% avec respectivement les caractéristiques linguistiques *CM*, *MH*, *Punct*, et *POS*. Cela est probablement dû au fait que ces informations sont déjà encodées dans les plongements de phrases de *USE*. Concernant notre système avec les plongements d'émojis générés par *emoji2vec*, nous observons qu'il obtient un score moyen *macro-F1* (72,3%) légèrement supérieur comparé au système de base (72,0%). Cependant, cette amélioration n'est pas significative. Nous observons que sur les ensembles de test de Davidson et Founta, l'utilisation des plongements d'émojis diminue légèrement les scores *macro-F1* comparé au système de base. Nous en concluons qu'ajouter les plongements des émojis en plus de leurs représentations sous forme textuelle n'est pas forcément utile pour aider le système de base sur la détection des discours haineux.

Caractéristiques	**Classification binaire**		**Classification ternaire**		**Moyenne**
	Waseem	**Hateval**	**Davidson**	**Founta**	
USE*	78,9 ($\pm 0,1$)	64,3 ($\pm 0,9$)	70,5 ($\pm 1,1$)	70,9 ($\pm 0,6$)	71,2
USE	79,5 ($\pm 0,1$)	63,9 ($\pm 0,4$)	72,2 ($\pm 0,5$)	72,4 ($\pm 0,7$)	72,0
USE - CM	79,1 ($\pm 0,4$)	65,3 ($\pm 0,6$)	71,1 ($\pm 0,7$)	72,2 ($\pm 0,3$)	71,9
USE - MH	78,4 ($\pm 0,5$)	63,2 ($\pm 3,6$)	**72,4** ($\pm 0,9$)	**72,8** ($\pm 0,3$)	71,7
USE - Punct	79,2 ($\pm 0,3$)	64,5 ($\pm 0,4$)	71,3 ($\pm 1,4$)	72,6 ($\pm 0,6$)	71,9
USE - EmoVec	**80,6** ($\pm 0,5$)	**65,2** ($\pm 0,4$)	71,7 ($\pm 1,2$)	71,6 ($\pm 0,4$)	**72,3**
USE - POS	78,8 ($\pm 0,2$)	65,2 ($\pm 0,6$)	70,9 ($\pm 1,7$)	71,3 ($\pm 0,6$)	71,6

TABLE 4.5 – Médianes des scores *macro-F1* et écart-types sur 5 entraînements. Dans la colonne *Caractéristiques*, le symbole "*" signifie que nous avons appliqué un pré-traitement spécifique supprimant totalement les émojis dans les tweets. Les résultats soulignés indiquent une amélioration significative par rapport au système de base (*USE*). Pour USE, les résultats soulignés représentent un amélioration significative par rapport au système *USE**. La colonne *Moyenne* représente la moyenne des scores médians sur les quatre jeux de données et l'amélioration significative est calculée en concaténant toutes les prédictions. Les meilleurs résultats sont en gras pour chacun des ensembles de test.

4.4.1 Statistiques des émojis dans les ensembles d'apprentissage de Waseem et HatEval

Ici, nous nous focalisons sur l'analyse des émojis contenus dans les ensembles d'apprentissage de Waseem et de HatEval afin de comprendre l'amélioration des résultats sur ces ensembles de test avec l'utilisation des émojis (sous forme textuelle et avec les plongements d'émojis).

Dans la figure 4.3, nous montrons la distribution (en pourcentage) des dix émojis qui occurrent le plus dans l'ensemble d'apprentissage de Waseem. Nous observons, dans cette figure, que les émojis 😂, 😳 et 🤦 sont plus souvent utilisés dans les tweets haineux par rapport à l'utilisation dans les tweets normaux avec respectivement 32,6%, 8,6% et 6,5% d'apparitions des ces émojis dans les tweets haineux contre 9,2%, 1,4% et 2,9% d'apparitions dans les tweets normaux. Nous remarquons aussi que certains émojis sont utilisés uniquement dans les tweets normaux, comme c'est la cas des émojis 😘 et 👏 qui apparaissent respectivement dans 13,2% et 4,7% des tweets normaux. Cela explique probablement l'augmentation des résultats entre les systèmes *USE** (qui n'utilise pas les émojis) et *USE* (qui utilise les émojis sous forme textuelle). Cela peut aussi expliquer l'augmentation des résultats entre les systèmes utilisant *USE* et *USE-EmoVec*, car le second système utilise en plus des émojis sous forme textuelle les plongements d'émojis générés par *emoji2vec*.

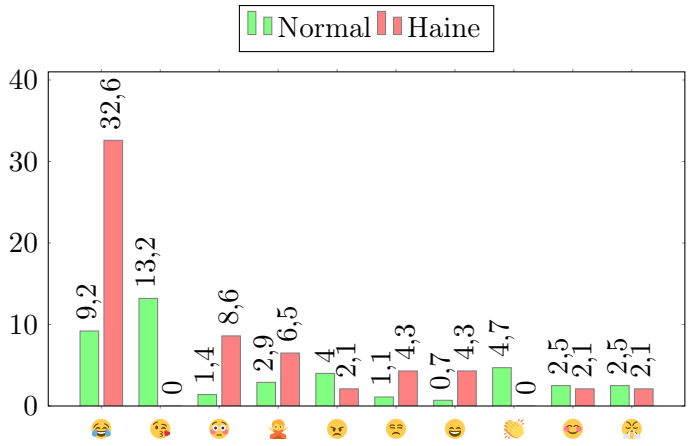


FIGURE 4.3 – Distribution (en pourcentage) des 10 émojis les plus présents dans l'ensemble d'apprentissage de Waseem.

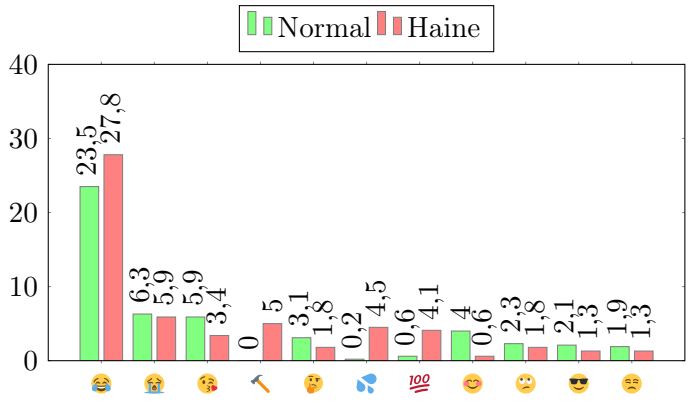


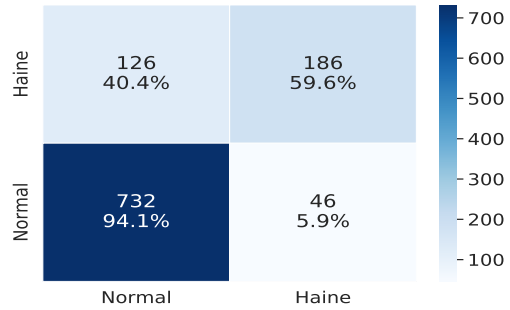
FIGURE 4.4 – Distribution (en pourcentage) des top 10 émojis qui apparaissent le plus dans l'ensemble d'apprentissage de HatEval.

Dans la figure 4.4, nous exposons la distribution des dix émojis les plus présents dans l'ensemble d'apprentissage de HatEval. Nous constatons, comme sur l'ensemble d'apprentissage de Waseem, que l'émoji 😂 est le plus présent et apparaît le plus fréquemment dans les tweets hai-

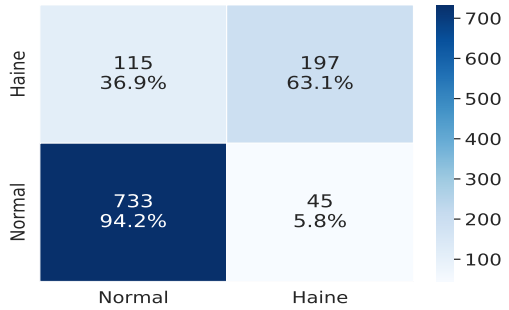
neux. En revanche, contrairement à l'observation sur l'ensemble d'apprentissage de Waseem, nous remarquons que sur HatEval les émojis apparaissent quasiment équitablement dans les tweets normaux et haineux, comme c'est le cas des émojis 😭, 😘, 😊, 🙄, 😎 et 😒. Cependant, nous constatons que les émojis 🔨 et 💦 apparaissent très majoritairement dans des tweets haineux. Pour l'émoji 🔨, nous avons constaté qu'il est utilisé pour exprimer la haine envers les immigrants en faisant référence à la construction du mur à la frontière des États-Unis et du Mexique. Concernant l'émoji 💦, il est souvent utilisé pour exprimer la haine envers les femmes avec une connotation sexuelle.

4.4.2 Matrices de confusion

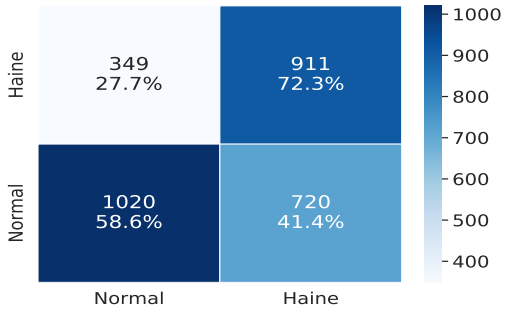
Dans cette section, nous comparons les matrices de confusion (présentées dans la figure 4.5) du système de base avec *USE** (figures 4.5(a) et 4.5(c)) et notre système avec les caractéristiques *USE-EmoVec* (figures 4.5(b) et 4.5(d)) sur les ensembles des test de Waseem et HatEval.



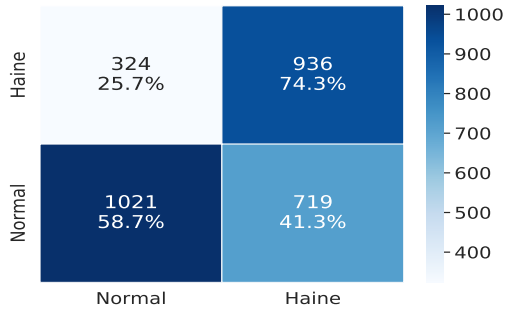
(a) Système de base avec *USE** sur Waseem



(b) Système proposé avec *USE-EmoVec* sur Waseem



(c) Système de base avec *USE** sur HatEval



(d) Système proposé avec *USE-EmoVec* sur HatEval

FIGURE 4.5 – Matrices de confusion sur les ensembles de test de Waseem et HatEval pour le système de base avec *USE** (a et c) et le système utilisant les émojis *USE-EmoVec* (b et d).

Concernant la comparaison des matrices de confusion sur l'ensemble de test de Waseem, nous constatons que le système avec les caractéristiques *USE-EmoVec* (figure 4.5(b)) corrige 3,5% des tweets haineux classifiés comme normaux par le système de base avec *USE**. De plus, nous observons que le système proposé avec *USE-EmoVec* ne dégrade pas la classification des tweets normaux comparé au système de base avec *USE** : 94,2% et 94,1% des tweets normaux correctement identifiés par respectivement les systèmes avec *USE-EmoVec* et *USE**. Nous en

concluons que la caractéristique des émojis aide à la détection des discours haineux dans les tweets.

Concernant les matrices de confusion sur HatEval (figures 4.5(c) et 4.5(d)), nous observons le même phénomène que sur l'ensemble de test de Waseem. Le système proposé avec *USE-EmoVec* améliore la classification des tweets haineux d'environ 2% sans dégrader la classification des tweets normaux comparé au système de base avec *USE**.

4.5 Conclusion du chapitre

Dans ce chapitre, nous avons étudié l'ajout de caractéristiques linguistiques à l'utilisation des plongements de phrases dans un système neuronal pour aider la détection des discours haineux. Nous avons proposé un système neuronal à deux branches utilisant les plongements de phrases et intégrant les caractéristiques linguistiques. Les caractéristiques linguistiques ajoutées sont les suivantes : les mots haineux, la casse des mots, la répétition de la ponctuation, les POS et les émojis. Nous avons constaté que les émojis aident à la détection des discours haineux. Nous avons constaté que cette caractéristique aide sous forme textuelle, comme le pré-traitement proposé dans le chapitre 3.1.5, et en ajoutant les plongements d'émojis générés avec la boîte à outils *emoji2vec*. Dans les chapitres suivants (6, 7 et 8) sur la détection des discours haineux, nous appliquerons le pré-traitement introduit dans le chapitre 3.1.5.

5

Identification des expressions polylexicales dans les tweets

Dans ce chapitre, nous présentons notre contribution concernant l'identification des expressions polylexicales (EP) dans les tweets. L'identification des EP vise à étiqueter les unités lexicales (UL) appartenant à une EP dans une phrase. Notre objectif est d'étudier la robustesse de systèmes d'identification d'EP sur des données de Twitter, afin d'annoter automatiquement les jeux de données de la haine et d'étudier l'impact des EP sur la détection de la parole haineuse (voir le chapitre suivant). Notre intuition est que les EP pourraient aider la tâche de la détection des discours haineux dans les tweets. Nous étudions la robustesse de deux systèmes existants : le premier est basé sur **des règles et un dictionnaire** et le second est fondé sur **des réseaux neuronaux**. Nous proposons aussi une nouvelle approche basée sur la combinaison des deux systèmes étudiés.

Ce chapitre s'articule de la façon suivante : nous définissons les EP et leurs propriétés linguistiques en section 5.1. Puis nous introduisons la tâche d'identification des EP en section 5.2, suivi d'une présentation des systèmes état de l'art en section 5.3. Ensuite, nous motivons et exposons la méthodologie proposée en section 5.4. Nous présentons les différentes configurations expérimentales en section 5.5, et nous analysons les résultats obtenus en section 5.6 suivi d'une analyse des prédictions pour les deux systèmes étudiés en section 5.7. Enfin, en section 5.8 nous présentons notre nouvelle approche. Ce chapitre reprend et développe les résultats que nous avons publié dans les conférences suivantes : *13th Language Resources and Evaluation Conference* (Zampieri et al., 2022b) et *29e Conférence sur le Traitement Automatique des Langues Naturelles* (Zampieri et al., 2022a).

5.1 Expressions polylexicales

5.1.1 Définition

Une expression polylexicale est définie par un groupe d'UL (deux lexèmes ou plus) dont le sens différent du sens littéral. Par exemple, l'EP "*a piece of cake*" signifie *quelque chose de très facile* (sens idiomatique de l'expression) contrairement au sens littéral qui signifie *une part de gâteau*. Sag et al. (2002) définissent les EP comme des "interprétations idiosyncratiques qui dépassent la frontière des mots", signifiant qu'on ne peut déduire le sens des EP à partir des UL qui les composent. Dans l'exemple "*It is important to have an umbrella because here* ***it is raining cats and dogs !***", on ne peut déduire le sens de l'EP ***it is raining cats and dogs***

(en français : ***il pleut des cordes***) à partir des UL qui la compose. Baldwin and Kim (2010) définissent les EP par des UL qui (1) peuvent être décomposées en plusieurs lexèmes, et (2) exposent une idiosyncrasie lexicale, morphologique, syntaxique, sémantique et/ou statistique.

— Idiosyncrasie lexicale : se définit par le fait qu'une EP peut être formée par un ou plusieurs composants n'appartenant pas au dictionnaire de la langue. C'est le cas dans l'exemple "*we created a dictionary by* ***ad hoc***[17] *collection.*" avec l'EP ***ad hoc*** (en français : "*le dictionnaire est créé par une collecte spécifique*").

— Idiosyncrasie morphologique : s'exprime par le fait d'un figement morphologique[18] d'un ou plusieurs termes d'une EP. Par exemple, dans l'EP *to* ***kill two birds with one stone*** (en français : *faire d'une pierre deux coups*) les UL *one stone* ne sont pas flexibles morphologiquement, sinon cela ne fait plus référence à l'EP.

— Idiosyncrasie syntaxique : se justifie par la juxtaposition d'UL syntaxiquement incorrectes, comme dans l'EP anglaise ***by and large*** qui est construite par une coordination d'un adjectif et d'une préposition.

— Idiosyncrasie sémantique : se définit par les EP dont le sens diffère des mots composant l'EP. C'est le cas de l'EP ***don't judge a book by its cover*** (en français : *ne pas se fier aux apparences*). Ce type d'expression résulte de la non-compositionnalité sémantique qui ne permet pas de déduire le sens à partir des mots composant l'EP.

— Idiosyncrasie statistique : se définie par des combinaisons de mots ou d'UL qui apparaissent fréquemment dans le même ordre, comme par exemple l'EP ***black and white*** qui statistiquement apparaît très régulièrement dans cette ordre (Baldwin and Kim, 2010).

Dans la littérature toutes les définitions s'accordent sur le fait que les EP sont plus qu'une collocation de mots et comportent des propriétés linguistiques supplémentaires importantes (définies en 5.1.2). De plus, les EP peuvent être réparties dans différentes catégories que nous présentons dans 5.1.3.

5.1.2 Propriétés linguistiques des EP

Ici nous définissons les différentes propriétés linguistiques qu'une EP peut posséder : la discontinuité, la non-compositionnalité, l'ambiguïté et la variabilité.

Discontinuité des EP

La discontinuité se définie par le fait qu'une UL peut apparaître entre les UL formant une EP. On peut observer ce phénomène dans l'exemple suivant : "*Carlos* ***make*** *a* ***presentation***" (en français : *Carlos fait une présentation*), où le mot *a* intervient entre les mots de l'EP ***make*** et ***presentation***.

Non-compositionnalité des EP

La non-compositionnalité se définit par le fait qu'on ne puisse interpréter une EP à partir des UL qui la composent. Par exemple, l'EP ***pull someone's leg*** se traduit *faire une blague à quelqu'un*. Un locuteur ne connaissant pas la signification de l'EP ne pourra pas la déduire à partir de ses composantes. Par exemple :

— ***Break a leg*** (en français : *Souhaiter bonne chance à quelqu'un*)

17. Expression latine qualifiant un acte spécialement fait pour une formalité déterminée.
18. Une UL dont la forme textuelle ne varie pas dans l'EP.

— ***Spill the beans*** (en français : ***Vendre la mèche*** = *Trahir un secret*)

Ambiguïté des EP

Généralement, l'ambiguïté est une difficulté à par entière dans le domaine du traitement automatique des langues. Cette difficulté impacte le traitement des EP, puisque nous pouvons faire le choix du sens littéral ou du sens de l'EP. L'ambiguïté est notamment présente et problématique pour les expressions idiomatiques, puisqu'elles dépendent du contexte de la phrase. Les exemples ci-dessous illustrent l'ambiguïté des EP.

— *I'm clumsy, I* ***spilled the beans*** *for the surprise party.*

- en français : *Je suis maladroit, j'ai* ***vendu la mèche*** *pour la fête surprise.*

— *I'm clumsy, I spilled the beans on the floor.*

- en français : *Je suis maladroit, j'ai renversé les haricots sur le sol.*

Variabilité des EP

Les EP peuvent apparaître sous des formes différentes (ex : ***made presentations***) de leurs formes canoniques (ex : ***make presentation***). On peut observer une grande variabilité concernant les EP verbales. Par exemple, l'EP *to* ***take shower*** (en français : ***prendre douche***) présente de larges possibilités de variation :

— *I* ***take*** *a* ***shower*** (en français : *Je* ***prends*** *une* ***douche***)

— *I* ***took*** *a* ***shower*** (en français : *J'ai* ***pris*** *une* ***douche***)

— *He* ***takes*** *a* ***shower*** (en français : *Il* ***prend*** *une* ***douche***)

— *He* ***takes*** *many* ***showers*** (en français : *Il* ***prend*** *beaucoup de* ***douches***)

5.1.3 Catégories des EP

Les EP sont répertoriées dans des catégories, par exemple une EP nominale est composé de noms communs (*tax payer*). Une EP *idiomatique* verbale se définit par le fait que la tête syntaxique du groupe d'UL est un verbe, par exemple ***kick the bucket*** (en français : ***casser sa pipe*** = mourir). La table 5.1 donne les différentes catégories des EP fortes et leurs étiquettes correspondantes. Pour rappel, l'explication de la différence entre une EP forte et une EP faible est présentée dans la section 3.2.1. Pour chaque catégorie d'EP, un exemple d'EP extrait de l'ensemble d'apprentissage du jeu de données Streusle (introduit dans le chapitre 3) est présenté. Nous ne détaillons pas les différentes catégories d'EP ici. Schneider et al. (2014) et Savary et al. (2017) décrivent les différentes catégories d'EP et leurs constructions. Les organisateurs des différentes campagnes d'évaluation PARSEME proposent un guide d'annotation des EP verbales [19].

5.2 Identification d'EP

Dans cette section, nous introduisons la tâche d'identification des EP. Nous focalisons nos recherches sur les tweets qui présentent un langage non standard de texte court pouvant comprendre des abréviations, des erreurs grammaticales et syntaxiques, des émojis, etc. A notre connaissance, peu de travaux de recherches ont été effectués pour l'identification des EP dans les tweets (Schneider et al., 2016).

19. https ://parsemefr.lis-lab.fr/parseme-st-guidelines/

Famille	Noms	Étiquettes	Exemples
Verbale	Verbe intrinsèquement adpositionnel	V.IAV	*stand for*
	Construction à verbe léger (total)	V.LVC.full	*have option*
	Construction à verbe léger causatif	V.LVC.cause	*give liberty*
	Idiomatique verbale	V.VID	*give a crap*
	Construction à verbe support (total)	V.VPC.full	*take off*
	Construction à verbe support (semi)	V.VPC.semi	*walk out*
Autre	Adjectivale	ADJ	*dead on*
	Adverbiale	ADV	*once again*
	Discours	DISC	*thank you*
	Nominale	N	*tax payer*
	Auxiliaire	AUX	*be suppose to*
	Conjonction de coordination	CCONJ	*and yet*
	Determinant	DET	*a lot*
	Marqueur d'infinitif	INF	*to eat*
	Adposition	P	*apart from*
	Phrase d'adposition (idiomatique)	PP	*on the phone*
	Pronom non possessif	PRON	*my self*
	Conjonction de subordination	SCONJ	*even if*
	Symbole	SYM	*A+*
	Interjection	INTJ	*lo and behold*

TABLE 5.1 – Liste des catégories d'EP fortes avec leurs abréviations et des exemples en anglais.

5.2.1 Définition de la tâche

Le but d'un système d'identification d'EP est d'associer à chaque mot d'une phrase une étiquette correspondant à : (a) l'UL appartient à une EP ou non, (b) si l'UL appartient à une EP, un identifiant commun à toutes les UL formant l'EP. La table 5.2 montre un exemple d'étiquettes pour une phrase. Dans cette exemple, le système a annoté correctement les EP ***have surgery***, qui possède l'identifiant 1 et ***ingrown toenail*** identifiée par l'étiquette "2".

Entrée	I	**had**	a	routine	**surgery**	for	an	**ingrown**	**toenail**	.
Sortie		1			1			2	2	

TABLE 5.2 – Exemple d'entrée et de sortie d'un système d'identification des EP. Dans l'exemple, les EP sont en gras.

5.2.2 Difficultés de la tâche

Sag et al. (2002) qualifient les EP de "***pain in the neck***" (en français : ***casse-pieds***). Les propriétés linguistiques des EP, décrites en 5.1.2, font que l'identification des EP est une tâche très difficile pour les systèmes automatiques (Constant et al., 2017). En effet, la discontinuité, la non-compositionnalité, l'ambiguïté et la variabilité posent des problèmes aux systèmes automatiques.

Les EP **discontinues** sont complexes à identifier du fait que des UL interviennent entre les mots composant une EP. De plus, il est difficile de déterminer pour un système le nombre d'UL qui peuvent intervenir entre les mots composant une EP.

La **non-compositionnalité** est une des propriétés qui pose le plus de problèmes aux systèmes

d'identification d'EP. En effet, il est difficile pour les systèmes automatiques de découvrir des EP jamais observées lors de l'apprentissage.

L'**ambiguïté** des EP pose aussi des problèmes. En effet, un groupe d'UL pouvant former une EP est dépendant du contexte dans la phrase. Par exemple, l'EP ***a piece of cake*** doit être identifié dans la phrase "Writing a thesis is ***a piece of cake*** !" (sens idiomatique de l'EP), mais pas dans la phrase "*I eat a piece of cake.*" (sens littéral du groupe d'UL).

La **variabilité** des EP pose un problème dû au fait que les UL d'une EP peuvent apparaître sous différentes formes dans le texte. De plus, l'invariance de certaines EP peuvent aussi poser des problèmes aux systèmes d'identification d'EP car ces EP ne doivent être identifiées uniquement lorsque les UL ont une forme précise.

En plus des difficultés liées aux propriétés linguistiques des EP, les tweets apportent une dimension plus complexe par rapport à l'identification des EP dans des textes standards. En effet, les tweets peuvent contenir des fautes d'orthographe (ex : ***fck up***), une syntaxe non-standard (ex : ***omg*** *but stop with articles about Lea and her love for Cory it hurts :'(and it's not a good source...*) et des erreurs de grammaire (ex : *Target :Tena* ***Money Maker Deal!!***). Les tweets peuvent aussi contenir des abréviations, comme "***NY***" (***New York***) ou bien "***STFU***" (***shut the f*ck up***).

5.3 Systèmes existants pour l'identification des EP

Constant et al. (2017) décrivent les différents travaux menés sur le traitement des EP et ils distinguent la découverte des EP de l'identification des EP. Ici, nous présentons un état de l'art des systèmes automatiques pour l'identification automatique des EP. Dans la littérature, on distingue deux méthodes majeures pour l'identification d'EP : Une approche basée sur des **règles** (section 5.3.1) ; et une méthode devenue très populaire récemment basée sur l'**apprentissage supervisé** (section 5.3.2).

5.3.1 Systèmes à base de règles

Une méthode simple pour identifier les EP serait d'étiqueter, à chaque fois comme EP, les UL d'une EP connue (ex : ***spill the beans***) dans des nouvelles phrases. Cependant, un système à base de règles peut être plus sophistiqué. Par exemple, les règles peuvent être représentées par des transducteurs à états finis (ou automates à états finis). Gross (1989) propose un dictionnaire à automate fini afin d'identifier les EP figées [20] dans les phrases.

Une autre approche repose sur l'utilisation d'expressions régulières pour représenter les règles. Breidt et al. (1996) ont établi un dictionnaire en ajoutant des expressions régulières et des contraintes pour différentes EP. Les auteurs donnent comme exemple l'expression régulière de l'EP française "***perdre la tête***" : *perdre*(V) <ADV>* [:*la* :*tête* | :*la* :*boule* | :*les* :*pédales*], où *perdre* est un verbe (V) qui peut être suivi d'un nombre non-spécifique d'adverbes (ADV). Les UL *la tête*, *la boule* et *les pédales* sont morphologiquement invariant.

Cependant, créer un tel dictionnaire de règles d'EP est très complexe. De ce fait, des systèmes à base de règles utilisant des dictionnaires plus simples ont émergé. Ces dictionnaires ne contiennent plus des règles d'EP, mais les formes canoniques des mots composant les EP sans ajout de contraintes (Carpuat and Diab, 2010; Ghoneim and Diab, 2013). Ces systèmes s'utilisent en deux étapes : (1) lemmatisation des phrases et annotation en parties du discours, et (2) utilisation d'algorithmes de recherche par correspondance exacte. Néanmoins, ces systèmes

20. EP ne présentant pas de variations des formes textuelles des mots.

à base de dictionnaires d'EP sans ajout de contraintes se retrouvent face à un problème : la non prise en charge de la discontinuité et de l'invariance de certaines EP. Par exemple, "*I'm clumsy, I spilled the beans on the floor.*", l'EP "***spill the beans***" sera annotée si l'EP est présente dans le dictionnaire, alors que dans cette exemple les UL *spill the beans* ne font pas références au sens idiomatique de l'EP.

Afin de pallier les problèmes d'ambiguïté et de discontinuité, il est possible de coupler l'utilisation d'un dictionnaire d'EP avec des heuristiques. Par exemple, la boîte à outils *mwetoolkit* (Ramisch et al., 2010) permet de paramétrer : le nombre d'UL pouvant apparaître entre les UL composant une EP, l'utilisation d'expressions régulières basées sur les parties du discours, etc. Cette boîte à outils a obtenu la deuxième place lors de la campagne d'évaluation DiMSUM (Cordeiro et al., 2016).

5.3.2 Systèmes à base d'apprentissage supervisé

Une autre approche pour l'identification des EP est d'utiliser l'apprentissage supervisé. L'apprentissage supervisé est basé sur la disponibilité de jeux de données annotés.

Constant and Nivre (2016) ont proposé un système à base des transitions pour l'annotation lexical et syntaxique des phrases. Un tel système , à chaque étape, effectue une transition vers un état, qui représente la structure syntaxique de la phrase à ce stade de l'analyse. Les transitions sont déterminées par un ensemble de règles syntaxiques préalablement définies. Le système est basé sur l'utilisation d'un *oracle* qui applique une règle unique pour passer d'un état à un autre. Maldonado et al. (2017) ont développé un système d'identification d'EP utilisant des champs aléatoires conditionnels (CRF pour *Conditional Random Fields*) qui prennent en compte différentes caractéristiques des mots : la forme textuelle, la forme canonique, la partie du discours et la relation de dépendance de l'arbre syntaxique. Waszczuk (2018) a développé un système, nommé *TRAVERSAL*, pour l'identification des EP à base de CRF arborescents. *TRAVERSAL* représente une phrase par un hyper-graphe (Klein and Manning, 2004) dans lequel chaque parcours de l'arbre de dépendance est représenté par un hyper-chemin distinct dans cet hyper-graphe. Dans cet hyper-graphe chaque noeud est représenté par des hyper-noeuds encodant l'information sur les EP. Par exemple, un hyper-noeud encode si lui-même appartient à une EP, si son parent dans le graphe appartient à une EP, etc. Ce système a obtenu la première place lors de la campagne d'évaluation PARSEME (en mode *close-track*[21]).

Avec les avancées réalisées dans le traitement automatique des langues, les systèmes à apprentissage supervisé basés sur les réseaux de neurones se sont rapidement développés comme en témoigne les systèmes soumis à la campagne d'évaluation PARSEME 1.1 en 2018 : 9 des 17 systèmes soumis sont basés sur des réseaux neuronaux (Berk et al., 2018; Boros and Burtica, 2018; Ehren et al., 2018; Taslimipoor and Rohanian, 2018; Stodden et al., 2018; Zampieri et al., 2018). Ces systèmes utilisent des plongements de mots (*word embeddings* en anglais) pour représenter les mots en entrée des systèmes. Taslimipoor and Rohanian (2018) ont développé *Shoma*, un système multi-tâches pour l'identification d'EP et la prédiction d'arbres syntaxiques, basé sur des réseaux de neurones récurents bidirectionnels (bi-RNN). Zampieri et al. (2018) ont proposé le système, *Veyn*, basé aussi sur des bi-RNN utilisant en entrée plusieurs caractéristiques pour représenter les mots : des plongements des formes textuelles et canoniques des mots et la partie du discours. Quand au système *Mumpitz*, proposé par Ehren et al. (2018), il utilise des bi-RNN couplés à l'utilisation d'heuristiques pour différencier des EP verbales dans la phrase : si un verbe est identifié comme appartenant à une EP, alors les mots directement dépendant de

21. Cela signifie que le système a utilisé uniquement les données fournies par les organisateurs de la campagne d'évaluation.

ce verbe sont étiquetés comme appartenant à la même EP et à l'inverse, si un verbe n'est pas étiqueté comme appartenant à une EP mais que des mots dépendant de ce verbe le sont, alors le verbe est considéré comme appartenant à l'EP. Berk et al. (2018) ont développé un système basé sur des bi-RNN suivi d'une couche de décision CRF pour l'identification d'EP. Boros and Burtica (2018) ont proposé un système, nommé *GBD-NER*, utilisant aussi des bi-RNN, mais encodant les phrases sous forme de graphes avec chaque noeud correspondant à un mot (comportant différentes informations, forme textuelle, canonique, partie du discours, etc.). Le but de *GBD-NER* est d'identifier les sous-graphes dans la phrase correspondant aux graphes des EP. Stodden et al. (2018) ont proposé les systèmes *TRAPACC* et *TRAPACCs*, lesquels reprennent l'idée d'un étiquetage des transitions de dépendance de Constant and Nivre (2016). *TRAPACC* et *TRAPACCs* utilisent un système de réseau neuronal convolutionnel pour extraire des caractéristiques à partir de phrases, puis un modèle de classification est entraîné pour prédire les transitions. Si une transition forme une entité nommée, alors les mots correspondant à cette transition sont annotés en tant qu'entité nommée.

Avec l'arrivée des modèles de langues pré-entraînés basés sur des *transformers*[22](Vaswani et al., 2017b), tel que BERT (Devlin et al., 2019) ou RoBERTa (Liu et al., 2019b), des systèmes basés sur les *transformers* ont émergé pour l'identification des EP. Cela est attesté par les deux meilleurs systèmes de la campagne d'évaluation PARSEME 1.2 (Ramisch et al., 2020) qui sont basés sur les *transformers*. Le système *MTLB-struct* (Taslimipoor et al., 2020) est basé sur le modèle BERT avec un entraînement multi-tâches comme proposé par Taslimipoor and Rohanian (2018). Le système *TRAVIS-multi* exploite un modèle de langue pré-entraîné sur des données multi-langues (Kurfalı, 2020). Liu et al. (2021) ont proposé un analyseur lexical basé sur un modèle BERT qui prédit les EP et les étiquettes de super-sens simultanément. Ce système a obtenu des résultats équivalant aux systèmes état de l'art sur le jeu de données Streusle (Schneider and Smith, 2015), sur PARSEME (Ramisch et al., 2018) et sur DiMSUM (Schneider et al., 2016).

5.4 Méthodologie proposée

Dans cette section, nous établissons notre méthodologie pour l'étude de la robustesse de deux systèmes existants d'identification des EP dans les tweets. L'objectif est de pouvoir annoter automatiquement des jeux de données de haine en termes d'EP afin d'étudier l'impact des EP sur la détection des discours haineux. Les deux systèmes d'identification d'EP étudiés sont de types différents : le premier est basé sur un **dictionnaire** et le second sur des **réseaux neuronaux**.

Concernant le système à base de dictionnaire, nous utilisons la boîte à outils *mwetoolkit* (Cordeiro et al., 2016). Le *mwetoolkit* facilite la création d'un dictionnaire d'EP et permet de projeter ce dictionnaire sur un nouveau texte en utilisant des règles. De plus, l'utilisation de cette boîte à outils à permis à Cordeiro et al. (2016) de terminer deuxième de la campagne d'évaluation DiMSUM en 2016. Nous développons l'utilisation de la boîte à outils dans la sous-section 5.4.1.

Pour le système neuronal, nous utilisons le système de reconnaissance lexical (LSR pour *Lexical Semantic Recognition*) (Liu et al., 2021). Nous avons choisi ce système car il a obtenu de bons résultats sur différents jeux de données. En effet, le système proposé par Liu et al. (2021) parvient à obtenir des résultats similaires à ceux des systèmes les plus avancés sur les ensembles de données de référence de Streusle, DiMSUM et PARSEME 1.1. De plus, ce système à l'avantage de ne pas nécessiter d'annotation morpho-lexicale, morpho-syntaxique ou sémantique des jeux de données. Cela représente un grand avantage pour annoter automatiquement en termes d'EP les jeux de données de la parole haineuse, car ils ne possèdent pas ce type d'annotation.

22. Réseaux neuronaux avec mécanisme d'auto-attention.

5.4.1 Système à base de dictionnaire

Le système à base de dictionnaire qui comporte deux étapes : la création d'un dictionnaire d'EP, puis la projection du dictionnaire sur une phrase.

Nous créons le dictionnaire à partir de jeux de données annotés manuellement. Chaque mot extrait des EP est lemmatisé et les formes canoniques des EP sont placées dans le dictionnaire d'EP. Le dictionnaire contient à la fois des EP comprenant des mots contigus et non-contigus [23] de jeux de données annotés. Concernant les EP non-contiguës, seuls les UL composant l'EP sont conservées, sans tenir compte des mots intermédiaires. Par exemple, en extrayant les EP de la phrase "*I **had** a routine **surgery** for an **ingrown toenail**.*", nous mettrons les formes canoniques ***have surgery*** et ***ingrown toenail*** dans le dictionnaire. La création du dictionnaire s'effectue une seule fois. La partie verte de la figure 5.1 montre la création du dictionnaire d'EP.

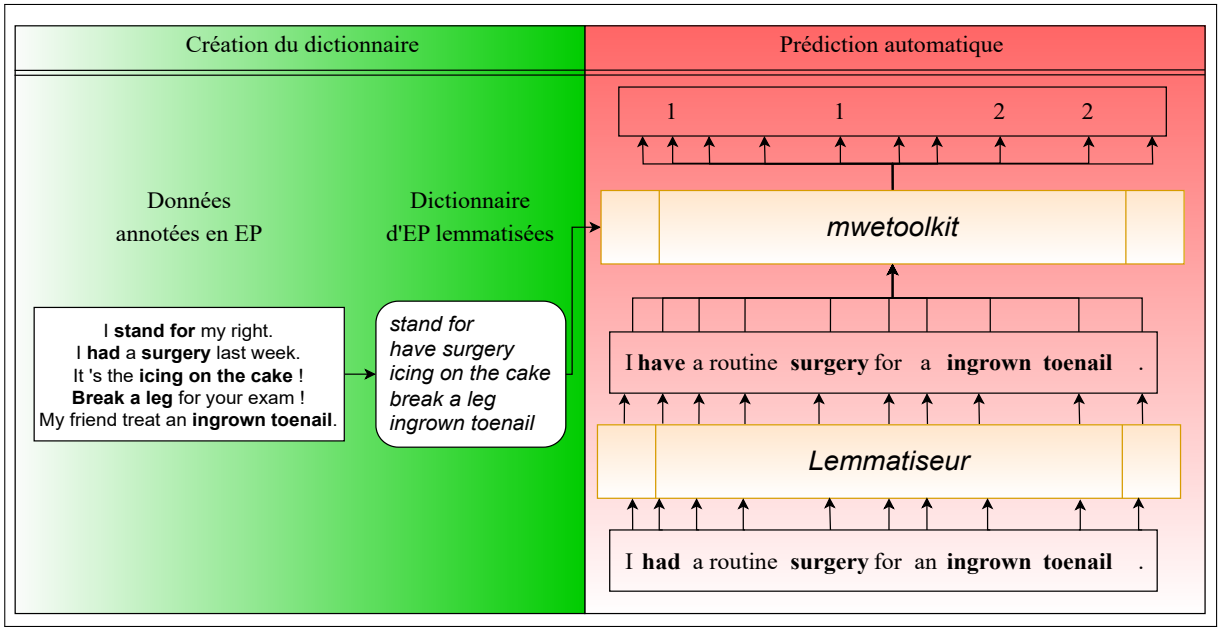


FIGURE 5.1 – Exemple d'utilisation de la boîte à outils *mwetoolkit* (Cordeiro et al., 2016) pour la création d'un dictionnaire d'EP et pour la prédiction d'EP.

Maintenant que le dictionnaire est créé, nous projetons les EP contenues dans le dictionnaire sur une nouvelle phrase. Il est important de noter que les mots de la phrase doivent être sous forme canonique afin de pouvoir projeter les EP. La projection se déroule par correspondance exacte entre les UL d'une EP et les UL présentes dans la phrase. Le système possède un paramètre pour la prise en compte de la discontinuité des EP en prévoyant qu'une ou plusieurs UL peuvent intervenir entre les UL d'une EP lors de la projection. Les parties du discours des UL peuvent aussi être utilisées en entrée du système en plus des formes canoniques des EP pour la projection des EP. La partie rouge de la figure 5.1 montre la projection par correspondance exacte des EP.

23. EP ayant fait preuve de discontinuité dans une phrase.

5.4.2 Système neuronal

Le système LSR est un système qui prend une phrase en entrée et prédit en sortie des étiquettes pour chacun des mots de la phrase. La figure 5.2 illustre l'architecture du système LSR. Ce système utilise le modèle BERT (Devlin et al., 2019), suivi de deux couches bi-LSTM et d'une couche CRF. Il est essentiel de noter que le modèle BERT est gelé lors de l'apprentissage. Seuls les poids des couches neuronales supérieures, bi-LSTM et CRF, sont ajustés. Dans le système LSR, les EP sont représentées avec un schéma d'étiquette "**BIO**" comme dans la figure 5.2. L'étiquette du premier mot composant une EP est "B" (*begin*), les mots suivants de l'EP sont étiquetés "I" (*inside*), et les mots n'appartenant à aucune EP sont étiquetés "O" (*ouside*).

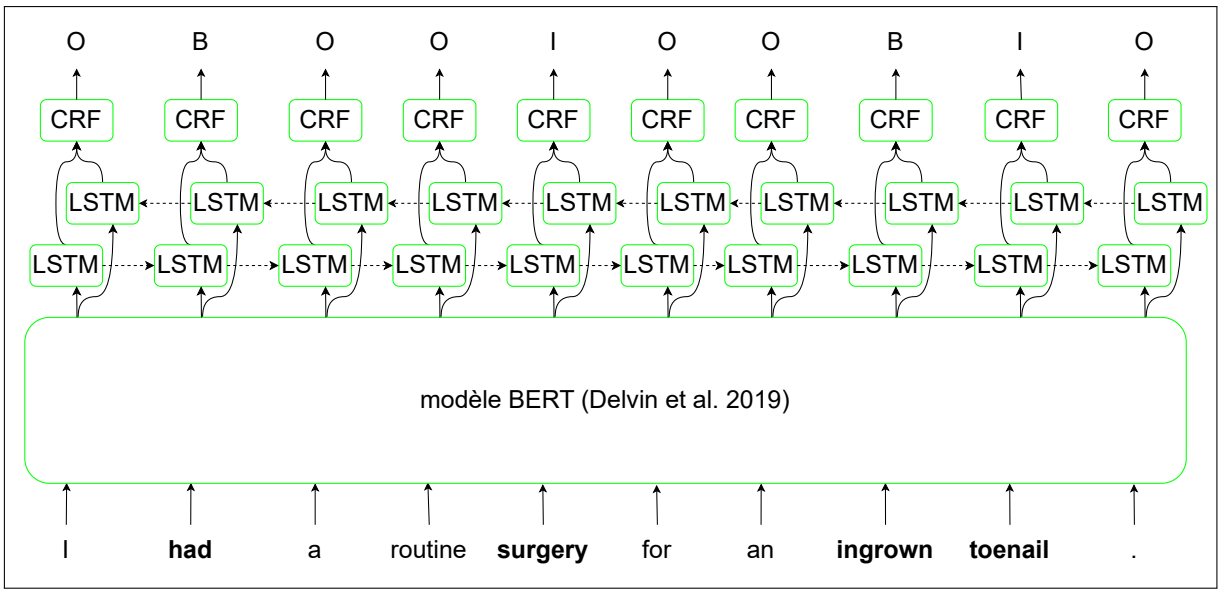


FIGURE 5.2 – Système automatique d'identification d'EP proposé par Liu et al. (2021), avec une phrase d'exemple en entrée et les étiquettes associées pour chacun des mots en sortie au format "BIO".

Schémas d'étiquettes											
	BIOo	O	B	O	O	I	O	O	B	I	O
	BIOo-cat	O	B-V.LVC.full	o	o	I-V.LVC.full	O	O	B-N	I-N	O
	BIOo-cat-ss	O-PRON	B-V.LVC.full.v.social	o-DET	o-ADJ	I_	O-P-p.purpose	O-DET	B-N-n.body	I_	O-PUNCT
	Phrase	I	**had**	a	routine	**surgery**	for	an	**ingrown**	**toenail**	.

FIGURE 5.3 – Exemple d'étiquetage "BIO" pour le modèle LSR. Cet exemple possède deux EP (en gras) : *had surgery* et *ingrown toenail*.

Nous évaluons trois schémas d'étiquetage différents exposés dans la figure 5.3. Le schéma d'étiquettes "**BIOo**" est fondé sur le schéma "BIO" standard. Il utilise une étiquette supplémentaire "o" qui signifie qu'une UL intervient entre les UL formant une EP. Par exemple, dans

la figure 5.3 les UL *a routine* interviennent entre les UL de l'EP **had surgery**. Ce schéma d'étiquetage contient uniquement 4 étiquettes.

Le schéma d'étiquettes "**BIOo-cat**" est une variante du schéma précédent "BIOo". Il consiste à ajouter à chaque étiquette "B" et "I" la catégorie de l'EP. Sachant qu'on utilise 20 catégories d'EP, ce schéma d'étiquetage contient 42 étiquettes.

Le schéma d'étiquettes "**BIOo-cat-ss**" correspond au schéma d'étiquetage proposé par Liu et al. (2021). Ce schéma d'étiquettes utilise, en plus des catégories des EP, le super-sens des mots. Pour rappel, le super-sens d'une UL correspond à sa classe sémantique : par exemple la classe sémantique de *toenail* est *n.body* (*partie du corps*). Il concatène aux étiquettes "B" les catégories des EP et les étiquettes de super-sens, aux étiquettes "I" un marqueur "_", et aux étiquettes "O" le super-sens. Il convient de noter qu'il utilise des étiquettes supplémentaires "b", "i" et "o" ayant la même signification que les étiquettes "B", "I" et "O", mais indiquant que les UL sont imbriquées dans une EP. Ce jeu d'étiquette contient 601 étiquettes avec plus de 220 étiquettes de super-sens.

Nous appliquons un filtrage sur la sortie du système LSR que nous détaillons plus tard. En plus d'évaluer différentes configurations du système LSR avec plusieurs schémas d'étiquettes, nous varions également les données d'apprentissage. Nous utilisons à la fois des tweets et des données qui ne sont pas des tweets. Cela est important pour déterminer si le système doit être entraîné avec des données du même domaine pour prédire correctement les EP dans les tweets. Dans la section suivante, nous fournissons plus d'informations sur les différentes configurations explorées pour le système LSR.

5.5 Conditions expérimentales

Dans cette section, nous décrivons les différentes configurations que nous avons explorées pour les systèmes à base de dictionnaire et de réseaux neuronaux. Pour rappel, les jeux de données (Streusle, DiMSUM et PARSEME), le pré-traitement, ainsi que les mesures d'évaluation utilisés pour l'identification des EP dans les tweets sont décrits dans le chapitre 3 (section 3.2). Afin de simplifier la compréhension des résultats, nous résumons les configurations décrites ci-dessous dans la table 5.3.

5.5.1 Configuration du système à base de dictionnaire

Pour la configuration du système à base de dictionnaire, nous extrayons les EP de tous les jeux de données annotés manuellement en EP, à l'exception de l'ensemble de test de DiMSUM car c'est l'ensemble qui nous sert de test. C'est-à-dire, nous récupérons les formes canoniques des EP des jeux de données de Streusle et PARSEME, et aussi de l'ensemble d'apprentissage de DiMSUM. Il est important de rappeler que tous les jeux de données (ensembles d'apprentissage, de validation et de test) utilisés sont lemmatisés manuellement. Le dictionnaire obtenu contient 3255 EP.

Nous utilisons l'ensemble d'apprentissage de DiMSUM pour régler le paramètre de l'écart maximal entre deux mots composant une EP (pour prendre en compte la discontinuité des EP) du système basé sur le dictionnaire. Nous sommes conscient que les EP ont été extraites de cet ensemble de données, cependant nous réglons ce paramètre sur le seul jeu de données disponible (hors ensemble de test) contenant des tweets. Nous avons expérimenté plusieurs valeurs pour ajuster l'écart maximal entre les mots composant les EP discontinues. Nous avons trouvé pour valeur optimale 3 mots maximum pouvant séparer deux UL d'une EP. Nous avons aussi évalué l'utilisation de l'étiquetage morpho-syntaxique (*partie du discours*) en entrée du système avec

les formes canoniques des mots composants une EP. L'étiquetage morpho-syntaxique n'a pas montré d'amélioration significative des résultats sur les tweets de l'ensemble d'apprentissage de DiMSUM. Donc, nous utilisons uniquement les formes canoniques des EP.

5.5.2 Configurations du système neuronal

Le système LSR utilise le modèle *BERT-large-cased* qui possède un espace de plongement de taille 1024 et qui est sensible à la casse des mots. Nous rappelons que le modèle BERT est gelé lors de l'apprentissage. Chaque couche de bi-LSTM a une taille de 512 neurones (256x2 neurones). Pour le système LSR, nous entraînons sept configurations différentes (1-7 ci-dessous). Pour chaque configuration, nous effectuons cinq apprentissages avec une initialisation aléatoire. Afin d'éviter le sur-apprentissage, nous utilisons un arrêt précoce de l'apprentissage avec une patience de 25 époques et un nombre maximal d'époques de 75. Nous rappelons que les configurations proposées diffèrent en ce qui concerne les données d'apprentissage et la granularité des étiquettes.

(1) La configuration $\mathbf{LSR_{ST.cat\text{-}ss}}$ correspond à la configuration proposée par Liu et al. (2021). Dans cette configuration, nous entraînons le modèle LSR sur le corpus d'apprentissage de Streusle (avec les EP faibles et fortes) utilisant le schéma d'étiquettes "BIO-cat-ss".

(2) La configuration $\mathbf{LSR_{ST.cat}}$ est également entraînée sur l'ensemble d'apprentissage de Streusle. Cependant, elle n'utilise pas les EP faibles et les étiquettes de super-sens. Cette configuration utilise le schéma d'étiquettes "BIOo-cat".

(3) La configuration $\mathbf{LSR_{ST}}$ est également entraînée sur l'ensemble d'apprentissage de Streusle. Comme la configuration $\text{LSR}_{ST.cat}$, cette configuration n'apprend pas à prédire les EP faible. Nous utilisons le schéma d'étiquettes "BIOo".

(4) La configuration $\mathbf{LSR_{DSM}}$ utilise l'ensemble d'apprentissage DiMSUM. Le jeu de données DiMSUM n'étant pas étiqueté en termes de catégories pour les EP, nous utilisons le schéma d'étiquetage "BIOo".

(5) La configuration $\mathbf{LSR_{ST\text{-}DSM}}$ est entraînée sur les ensembles d'apprentissage de DiMSUM (tweets) et de Streusle (non-tweets). Comme les configurations précédentes ($\text{LSR}_{ST.cat}$, LSR_{ST} et LSR_{DSM}), cette configuration n'utilise pas les EP faibles contenues dans Streusle car le jeu de données DiMSUM est uniquement annoté en termes d'EP fortes. Nous utilisons les étiquettes "BIOo", pour la même raison que la configuration précédente LSR_{DSM}.

(6) La configuration $\mathbf{LSR_{ST\text{-}PSM.cat}}$ est l'union de deux modèles qui utilisent le schéma d'étiquetage "BIOo-cat". Le premier modèle est entraîné sur les ensembles d'apprentissage de PARSEME et de Streusle, et prédit uniquement les EP verbales (14 étiquettes). Le second modèle est entraîné sur l'ensemble d'apprentissage de Streusle pour prédire les EP non-verbales (30 étiquettes). Lors de l'union des prédictions des deux modèles, si des EP se chevauchent ou sont imbriquées, nous choisissons de conserver l'EP qui apparaît en premier. Comme dans les configurations précédentes, celle-ci n'utilise pas les EP faibles.

(7) La configuration $\mathbf{LSR_{ST\text{-}PSM}}$ est identique à celle de $\text{LSR}_{ST-PSM.cat}$, à l'exception du schéma d'étiquettes utilisé. Dans cette configuration nous adoptons le schéma d'étiquettes "BIOo".

Pour évaluer l'impact de la granularité des étiquettes, nous comparons des configurations du système LSR qui sont entraînées sur le même ensemble d'apprentissage mais qui utilisent différents schémas d'étiquetage. Nous comparons ainsi les configurations $\text{LSR}_{ST.cat-ss}$, $\text{LSR}_{ST.cat}$ et LSR_{ST} qui sont entraînées sur l'ensemble d'apprentissage de Streusle avec des schémas d'étiquettes différents. De même, nous comparons les configurations $\text{LSR}_{ST-PSM.cat}$ et LSR_{ST-PSM},

qui sont entraînées sur les ensembles d'apprentissages de Streusle et PARSEME, en utilisant également différents schémas d'étiquetage. Pour évaluer l'impact des différents jeux de données sur l'identification dans les tweets, nous comparons les configurations du système LSR qui utilisent le même schéma d'étiquetage et qui sont entraînées sur différents jeux de données. Nous utilisons ainsi les configurations LSR_{ST}, LSR_{DSM}, LSR_{ST-DSM} et LSR_{ST-PSM}, qui utilisent toutes le schéma d'étiquettes "BIOo". Ces configurations nous permettent également de déterminer s'il est nécessaire d'entraîner le système LSR sur des données provenant uniquement de tweets, ou s'il est important d'utiliser des données issues d'autres sources de médias, ou encore une combinaison des deux.

Pour chaque configuration LSR décrite précédemment, nous utilisons l'ensemble de validation de Streusle pour ajuster les poids du modèle lors de l'apprentissage. En ce qui concerne le filtrage des sorties du LSR, nous avons évalué plusieurs heuristiques et avons retenu les suivantes : nous supprimons les EP prédites à UL unique, les étiquettes "I" qui ne sont pas précédées d'un "B", ainsi que les EP contenant des UL spéciales (@USER, URL et hashtags). La longueur maximale de l'écart entre les EP discontinues a également été ajustée et fixée à 2. Nous supprimons donc toutes les EP prédites dont l'écart entre les mots de l'EP est strictement supérieurs à 2.

Configurations	**Ensembles d'apprentissage**	**Schémas d'étiquettes**	**Nombre d'étiquettes**
Dictionnaire	Streusle*, PARSEME* & DiMSUM	-	-
$\text{LSR}_{ST.cat-ss}$	Streusle	BIO-cat-ss	601
$\text{LSR}_{ST.cat}$	Streusle	BIOo-cat	42
LSR_{ST}	Streusle	BIOo	4
LSR_{DSM}	DiMSUM	BIOo	4
LSR_{ST-DSM}	Streusle & DiMSUM	BIOo	4
$\text{LSR}_{ST-PSM.cat}$	Streusle & PARSEME	BIOo-cat	42
LSR_{ST-PSM}	Streusle & PARSEME	BIOo	4

TABLE 5.3 – Résumé des différentes configurations expérimentales des systèmes à base de dictionnaire et de réseaux de neurones, utilisées pour l'identification des EP dans les tweets. Le symbole * signifie que tous les ensembles du jeu de données sont utilisées pour l'apprentissage. Les diminutifs *ST*, *DSM* et *PSM* signifient respectivement Streusle, DimSUm et PARSEME.

5.6 Résultats obtenus

Dans cette partie, nous présentons les résultats de l'identification des EP sur le jeu de données de test de DiMSUM. La Table 5.4 présente les résultats pour les mesures *MWE-based*, *Token-based*, et *MWE-link-based*. La Table 5.5 expose les résultats obtenus pour les mesures d'évaluation *F-Non-vue*, *F-Variante*, et *F-Identique*.

5.6.1 Comparaison des systèmes à base de dictionnaire et de réseaux neuronaux

Dans cette partie, nous comparons les performances des systèmes d'identification des EP basés sur le dictionnaire et sur les réseaux de neurones. La table 5.4 montre que le système à base de dictionnaire atteint 28,7% pour le score F1 *MWE-based*. Cette performance est assez faible et s'explique par le fait que 78% des EP de l'ensemble de test ne sont pas présentes dans le

Configurations	MWE-based			Token-based	MWE-link-based
	Précision	**Rappel**	**score F1**	**score F1**	**score F1**
Dictionnaire	45,5	21,0	28,7	28,5	25,9
$LSR_{ST.cat-ss}$	45,5 ($\pm 3,4$)	29,9 ($\pm 2,0$)	36,1 ($\pm 2,4$)	47,6 ($\pm 1,3$)	43,8 ($\pm 1,4$)
$LSR_{ST.cat}$	53,7 ($\pm 1,1$)	36,4 ($\pm 2,6$)	43,3 ($\pm 1,6$)	53,5 ($\pm 2,1$)	51,2 ($\pm 2,1$)
LSR_{ST}	49,0 ($\pm 2,7$)	39,2 ($\pm 4,1$)	43,3 ($\pm 1,5$)	54,7 ($\pm 1,8$)	52,0 ($\pm 2,0$)
LSR_{DSM}	**61,1** ($\pm 2,7$)	31,2 ($\pm 2,6$)	41,2 ($\pm 2,3$)	51,8 ($\pm 3,1$)	48,5 ($\pm 2,7$)
LSR_{ST-DSM}	60,4 ($\pm 2,5$)	37,9 ($\pm 0,9$)	**46,5** ($\pm 0,3$)	**56,8** ($\pm 1,0$)	**54,0** ($\pm 1,5$)
$LSR_{ST-PSM.cat}$	53,2 ($\pm 1,5$)	37,1 ($\pm 2,0$)	43,6 ($\pm 1,3$)	54,1 ($\pm 1,7$)	50,9 ($\pm 1,7$)
LSR_{ST-PSM}	50,0 ($\pm 4,4$)	**39,9** ($\pm 3,3$)	44,1 ($\pm 0,9$)	54,7 ($\pm 2,1$)	51,9 ($\pm 2,4$)

TABLE 5.4 – Résultats d'identification d'EP dans les tweets de test de DiMSUM. Pour chaque résultats, le score moyen et l'écart type de 5 apprentissages sont donnés, sauf pour la configuration à base de dictionnaire (*Dictionnaire*).

Configurations	MWE-based					
	F-Non-vue		**F-Variante**		**F-Identique**	
	score F1	%	score F1	%	score F1	%
Dictionnaire	0,0	78%	50,9	8%	69,5	14%
$LSR_{ST.cat-ss}$	28,5 ($\pm 2,7$)	85%	61,8 ($\pm 5,6$)	5%	75,4 ($\pm 3,5$)	10%
$LSR_{ST.cat}$	37,4 ($\pm 2,3$)		60,1 ($\pm 5,6$)		76,0 ($\pm 2,8$)	
LSR_{ST}	36,5 ($\pm 2,2$)		66,1 ($\pm 5,4$)		**79,3** ($\pm 3,5$)	
LSR_{DSM}	40,6 ($\pm 2,4$)	87%	44,6 ($\pm 2,9$)	6%	47,1 ($\pm 6,4$)	7%
LSR_{ST-DSM}	**43,2** ($\pm 0,6$)	79%	55,9 ($\pm 3,7$)	8%	60,8 ($\pm 4,8$)	13%
$LSR_{ST-PSM.cat}$	36,7 ($\pm 1,1$)	85%	68,5 ($\pm 1,8$)	5%	78,1 ($\pm 4,1$)	10%
LSR_{ST-PSM}	37,1 ($\pm 1,3$)		**68,8** ($\pm 3,4$)		78,6 ($\pm 2,7$)	

TABLE 5.5 – Résultats d'identification d'EP sur les mesures d'évaluation *F-non-vue*, *F-Variante* et *F-Identique*. Pour chaque résultat, le score moyen et l'écart type de 5 apprentissages sont donnés, sauf pour la configuration à base de dictionnaire (*Dictionnaire*). La colonne % indique le pourcentage des EP, présentes dans l'ensemble de test, qui sont considérées comme *non-vues*, *variantes* ou *identiques* par rapport à l'ensemble d'apprentissage.

dictionnaire comme le montre la table 5.5 (ligne *Dictionnaire*). De plus, dans la table 5.4, nous remarquons que toutes les configurations du système LSR surpassent l'approche à base de dictionnaire. Nous observons que toutes les configurations LSR (hormis la configuration $LSR_{ST.cat-ss}$) améliorent le rappel et la précision (en termes de *MWE-based*) par rapport à l'approche à base de dictionnaire. Cela s'explique par le fait que les configurations LSR ont une meilleure capacité de généralisation, et surtout elles sont capable d'identifier des EP qui ne sont pas présentes dans l'ensemble d'apprentissage. La configuration du LSR_{ST-DSM} obtient les meilleurs résultats en termes de *MWE-based*, *Token-based* et *MWE-link-based* avec respectivement 46,5%, 56,8% et 54,0% de score F1.

Nous observons dans la table 5.5 que le système à base de dictionnaire surpasse les configurations LSR_{DSM} et LSR_{ST-DSM} pour identifier les EP qui apparaissent identiquement dans les ensembles d'apprentissage et de test (69,4% de F-identique contre 47,1% et 60,8%). De plus, le système à base de dictionnaire obtient un score plus élevé sur la mesure *F-Variante* comparé à la configuration LSR_{DSM} (50,9% contre 44,6%). Cela est sûrement dû au fait que la configura-

tion LSR_{DSM} est entraînée sur un petit nombre de tweets (moins de 1000 tweets) et qu'elle n'a observé que 7% des EP contenues dans le test lors de l'apprentissage.

Donc, nous pouvons conclure que le système à base de dictionnaire est moins performant que le système neuronal LSR et cela est surtout dû au fait que la majorité des EP dans l'ensemble de test ne sont pas présentes dans les ensembles d'apprentissage.

5.6.2 Impact des schémas d'étiquettes sur le système neuronal

Maintenant, nous nous focalisons sur l'étude de l'impact des schémas d'étiquettes pour le système LSR. En effet, il est essentiel d'étudier si les étiquettes de super-sens et les catégories des EP aident à l'identification des EP dans les tweets.

Nous comparons trois configurations $\text{LSR}_{ST.cat-ss}$, $\text{LSR}_{ST.cat}$ et LSR_{ST} entraînées sur le même ensemble d'apprentissage avec des schémas d'étiquettes différents : "BIO-cat-ss", "BIOo-cat" et "BIOo". Dans la table 5.4, nous observons qu'un jeu d'étiquette complexe ("BIO-cat-ss") diminue les scores F1. En effet, le système $\text{LSR}_{ST.cat-ss}$ obtient 36,1% de *MWE-based* contre 43,3% atteint par les systèmes $\text{LSR}_{ST.cat}$ ou LSR_{ST}. Cela s'explique par le fait qu'il est plus complexe de prédire correctement les EP pour une configuration contenant beaucoup d'étiquettes ("BIOo-cat-ss" contient 600 étiquettes possibles) qu'une configuration en contenant beaucoup moins : 42 et 4 étiquettes possibles respectivement pour les schémas d'étiquettes "BIOo-cat" et "BIOo". En effet, un système neuronal a plus de difficulté à généraliser avec 600 étiquettes comparé à 42 ou 4 étiquettes en utilisant la même quantité de données. Pour nos configurations, nous en déduisons que les étiquettes de super-sens n'aident pas à l'identification des EP dans les tweets. Nous observons la même performance pour les configurations utilisant le schéma "BIOo-cat" ou "BIOo" indiquant que les catégories des EP n'aident pas le système à mieux identifier les EP. Ce constat est confirmé avec les résultats obtenus par les systèmes $\text{LSR}_{ST-PSM.cat}$ et LSR_{ST-PSM} avec respectivement 43,6% et 44,1% de score F1 *MWE-based*. Nous constatons qu'un schéma d'étiquettes simple comme le "BIOo" (contenant seulement 4 étiquettes) obtient les meilleurs résultats comparé à ses homologues "BIOo-cat-ss" et "BIOo-cat" sur l'identification des EP dans les tweets. Sur l'ensemble de test utilisé, ajouter des informations lexicales et sémantiques, comme les catégories des EP et les étiquettes de super-sens, n'aident pas à mieux prédire les EP dans les tweets. En effet, nous observons avec les mesures *Token-based* et *MWE-link-based* le même constat : la configuration LSR_{ST} obtient de meilleurs résultats (54,7% de *Token-based* et 52,0% de *MWE-link-based*) que ses homologues $\text{LSR}_{ST.cat}$ (53,5% de *Token-based* et 51,2% de *MWE-link-based*) et $\text{LSR}_{ST.cat-ss}$ (47,6% de *Token-based* et 43,8% de *MWE-link-based*)

Dans la table 5.5, nous constatons que la configuration $\text{LSR}_{ST.cat-ss}$ obtient un moins bon score F1 sur la mesure *F-Non-vue* comparé à ses homologues $\text{LSR}_{ST.cat}$ et LSR_{ST} (28,5% contre 37,4% et 36,5% respectivement). Cela confirme que le système LSR a plus de difficulté à généraliser avec un schéma d'étiquettes complexe comme celui proposé par Liu et al. (2021). Nous observons que les configurations $\text{LSR}_{ST.cat-ss}$ et $\text{LSR}_{ST.cat}$ obtiennent des résultats similaires pour les mesures *F-Variante* (respectivement 61,8% et 60,1%) et *F-Identique* (respectivement 75,4% et 76,0%). Nous remarquons dans la table 5.5 que la configuration LSR_{ST} surpasse les configurations $\text{LSR}_{ST.cat-ss}$ et $\text{LSR}_{ST.cat}$ sur les mesures *F-Variante* (plus de 5% d'écart relatif) et *F-Identique* (plus de 3% d'écart relatif). Cependant, les configurations $\text{LSR}_{ST.cat}$ et LSR_{ST} arrivent à généraliser sur les EP jamais vue lors de l'apprentissage avec un score F1 *F-Non-vue* de 37,4% et 36,5%. Nous observons un constat similaire pour les configurations $\text{LSR}_{ST-PSM.cat}$ et LSR_{ST-PSM}.

Nous concluons que les catégories des EP et les étiquettes de super-sens n'aident pas à

identifier les EP dans les tweets.

5.6.3 Impact des jeux d'apprentissage sur le système neuronal

Maintenant, nous nous concentrons sur les configurations exploitant le même schéma d'étiquettes ("BIOo") mais différents ensemble d'apprentissage : LSR_{ST}, LSR_{DSM}, LSR_{ST-DSM} et LSR_{ST-PSM}. En effet, nous voulons étudier s'il est important que le système LSR soit appris sur des données de tweets, non tweets ou bien sur une combinaison de ces deux types de données.

Nous observons que la configuration LSR_{DSM}, entraînée uniquement sur des tweets, a les scores F1 les plus bas, atteignant 41,2% de *MWE-based*. Cela peut être dû au fait qu'il est entraîné avec le plus petit ensemble d'apprentissage, contenant 987 phrases, par rapport aux configurations LSR_{ST}, LSR_{ST-DSM} et LSR_{ST-PSM} qui contiennent plus de 2724 phrases. Le système LSR_{ST-PSM}, qui est entraîné sur les ensembles d'apprentissage de Streusle et de PARSEME, n'améliore pas les résultats, en termes de score F1, par rapport à la configuration LSR_{ST}, qui utilise uniquement Streusle. Cela peut être dû au fait que le système LSR_{ST-PSM} utilise deux modèles de réseaux neuronaux entraînés indépendamment. La configuration LSR_{ST-DSM}, qui est entraîné sur les ensembles d'apprentissage de Streusle et de DiMSUM, obtient les meilleurs scores F1 avec respectivement 46,5%, 56,8% et 54,0% de *MWE-based*, de *Token-based* et de *MWE-link-based*. Cela peut être dû au fait que le système LSR_{ST-DSM} est entraîné sur des données de tweets et de non-tweets.

Dans la table 5.5, nous constatons que la configuration LSR_{ST-DSM} obtient le plus haut score de *F-Non-vue* avec 43,2%. Il est important de noter que la configuration LSR_{DSM} atteint 40,6% de *F-Non-vue* et elle est la deuxième meilleure configuration pour généraliser sur des tweets. Cela confirme qu'il est important pour le système LSR d'apprendre sur des données de tweets. Cependant, l'utilisation des données de tweets et non tweets pour l'apprentissage du système LSR diminue les résultats pour les scores *F-Variante* et *F-Identique* : la configuration LSR_{ST} obtient 66,1% de *F-Variante* et 79,3% de *F-Identique* contre 55,9% de *F-Variante* et 60,8% de *F-Identique* pour la configuration LSR_{ST-DSM}. Nous remarquons, en comparant les configurations LSR_{ST} et LSR_{ST-PSM}, que le fait d'augmenter les données d'apprentissage de Streusle avec l'ensemble d'apprentissage de PARSEME ne donne aucune amélioration significative pour les scores *F-Non-vue* (36,5% contre 37,1%), *F-Variante* (66,1% contre 68,8%) et *F-identique* (79,3% contre 78,6%).

Nous pouvons conclure qu'apprendre le système LSR sur des données de tweets et de non-tweets est important pour l'identification des EP dans les tweets.

5.7 Analyse des erreurs

Dans cette section, nous analysons les erreurs de prédiction des systèmes basés sur des dictionnaires et des réseaux neuronaux. En ce qui concerne le système LSR, nous nous concentrons sur l'analyse de la configuration ayant obtenu le score F1 le plus élevé pour l'identification des EP : LSR_{ST-DSM}. Notre objectif est d'examiner les prédictions et les erreurs commises par les deux approches. Pour cela, nous commençons par analyser les prédictions de l'approche basée sur le dictionnaire, puis celles du système LSR_{ST-DSM}.

5.7.1 Erreurs du système à base de dictionnaire

L'erreur la plus fréquente observée dans le système basé sur le dictionnaire est l'incapacité à prédire les EP qui ne sont pas présentes dans le dictionnaire. En effet, l'approche par dictionnaire

Tweet	Tryna	go	to	**at**	**least**	5	phillies	games	**this**	**season**
Attendu	0	0	0	B	I	0	0	0	B	I
Prédit	0	B	I	B	I	0	0	0	O	O

Tweet	How	to	Find	And	Cultivate	Your	Passion
Attendu	O	O	O	O	O	O	O
Prédit	B	I	O	O	O	O	O

TABLE 5.6 – Exemples de tweets de l'ensemble de test de DiMSUM avec les EP *attendues* et *prédites* (au format "BIO") par le système à base de dictionnaire. Dans les exemples, les EP à prédire sont en **gras**.

ne peux pas identifier des EP qui ne sont pas présentes dans le dictionnaire. Nous avons également observé un autre type d'erreur : les insertions. Le système basé sur le dictionnaire commet des insertions, car il ne prend pas en compte le contexte de la phrase. La table 5.6 présente deux exemples d'EP insérées par erreur par le système basé sur le dictionnaire. Les insertions sont une source importante d'erreurs. Sur les 167 EP prédites par le système, 91 ont été insérées par erreur. Par exemple, l'EP ***go to***, qui dépend du contexte de la phrase (ex : *We are* ***going to*** *make beautiful babies.*), est étiquetée comme EP chaque fois qu'elle apparaît dans une phrase (premier exemple de la table 5.6). Nous observons la même erreur pour l'EP ***how to*** (ex : *I'm already now capable of picking up new songs off YouTube guitar* ***how to*** *videos*), qui dépend également du contexte de la phrase.

5.7.2 Erreurs du système neuronal

Dans cette section, nous analysons dles prédictions effectuées par la configuration LSR_{ST-DSM} qui est la configuration du système LSR ayant obtenu les meilleurs résultats en terme de *MWE-based*. Nous utilisons l'entraînement de la configuration LSR_{ST-DSM} ayant obtenu le score F1 médian en terme de mesure *MWE-based*. Pour rappel, nous effectuons un filtrage des EP sur la prédiction du système (introduit en 5.5.2) réduisant les erreurs produites par le système LSR. La table 5.7 montre deux exemples d'erreurs dans les prédictions du système LSR_{ST-DSM}. Il est important de noter que les deux exemples de la table 5.7 contiennent les EP ***happy Valentine's Day*** et ***Super Bowl*** qui ne sont pas observées lors de l'apprentissage.

Dans le premier exemple de la table 5.7, nous observons que le mot *happy* n'a pas été prédit comme appartenant à l'EP. Dans le second exemple, c'est le mot *Countdown* qui a été étiqueté comme appartenant à l'EP ***Super Bowl***. Nous avons observé que deux des entraînements du LSR_{ST-DSM} sur cinq ont réussi à prédire correctement ***happy Valentine's Day***. Au contraire, aucun des entraînements de cette configuration n'a réussi à prédire correctement l'EP ***Super Bowl***. Pour ces deux exemples, nous avons vérifié si ces erreurs était dû à la sensibilité à la casse du modèle BERT utilisé (*BERT-large-cased*), car le système prédit *Countdown* appartenant à une EP et non *happy* qui n'est pas en capitalisation. Nous avons constaté aucun changement des prédictions avec les phrases *Happy Valentine 's Day ! :) <3* et *Super Bowl countdown.*

5.8 Approche proposée en deux étapes

Dans cette section, nous proposons une nouvelle approche pour l'identification des EP dans les tweets. Notre nouvelle approche s'effectue en deux étapes. Elle est basée à la fois sur un système utilisant les réseaux de neurones et un dictionnaire. Le but est d'utiliser la robustesse de

Tweet	**happy**	**Valentine**	**s**	**Day**	!	:)	<3
Attendu	B	I	I	I	O	O	O
Prédit	O	B	I	I	O	O	O

Tweet	**Super**	**Bowl**	Countdown
Attendu	B	I	O
Prédit	B	I	I

TABLE 5.7 – Exemples de tweets de l'ensemble de test de DiMSUM avec les EP *attendues* et *prédites* (au format "BIO") par le système LSR_{ST-DSM}. Dans les exemples, les EP à prédire sont en gras.

chacun des deux systèmes. Le système basé sur les réseaux de neurones peut généraliser les EP sur de nouvelles phrases. Le système à base de dictionnaire permet une homogénéité des prédictions des EP avec la projection d'un dictionnaire par correspondance exacte. Nous supposons que le système à base de réseaux de neurones découvre des EP jamais observées lors de l'apprentissage ce qui donnerait un avantage au système à base de dictionnaire pour identifier les EP présents dans un lexique d'EP. Le risque de notre approche est que le système neuronal identifie des EP fausses ou partielles et que le système à base de dictionnaire annote ces EP dans les phrases. Cependant, Savary et al. (2019) soutiennent que les dictionnaires sont nécessaires pour obtenir une meilleure généralisation de l'identification d'EP.

La figure 5.4 illustre notre approche en deux étapes. Dans un premier temps, nous utilisons un système d'identification d'EP pré-entraîné pour prédire les EP dans un jeu de données cible (partie verte de la figure 5.4). Dans un second temps, nous augmentons, en termes d'EP, un dictionnaire d'EP en extrayant les EP prédites par le système neuronal que nous utilisons ensuite pour étiqueter le jeu de données cible (partie bleue dans la figure 5.4).

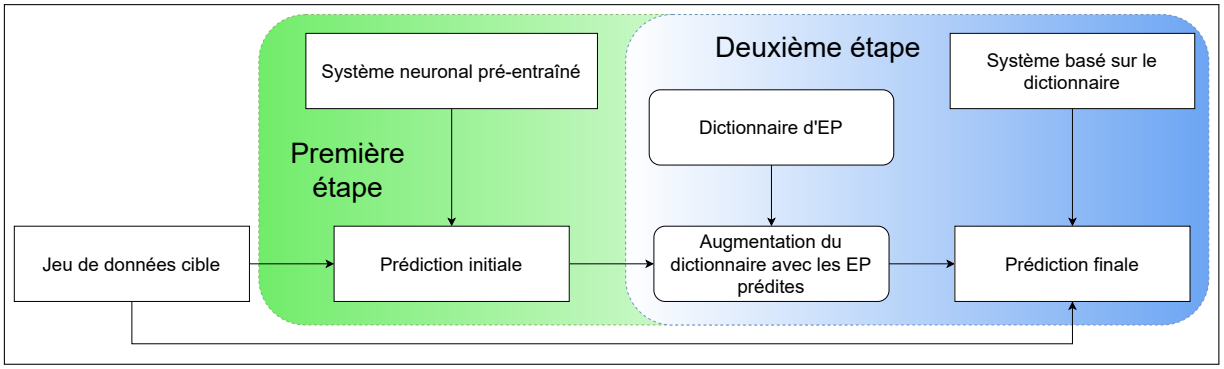


FIGURE 5.4 – Proposition d'approche pour l'identification des EP combinant le système à base de réseaux de neurones et de dictionnaire.

5.8.1 Configuration expérimentale

Nous prenons les systèmes étudiés précédemment : le système à base de dictionnaire et le système neuronal LSR_{ST-DSM}. Pour la configuration du système à base de dictionnaire, nous utilisons le dictionnaire créé précédemment et introduit dans section 5.5. Pour rappel, le dictionnaire est créé automatiquement en extrayant les EP de jeux de données annotés manuellement et il contient 3255 EP.

Concernant le système LSR, nous utilisons la configuration ayant obtenu le meilleur score sur les tweets : la configuration LSR_{ST-DSM}. Pour rappel, la table 5.3 énumère les différentes configurations étudiées précédemment. La configuration du LSR_{ST-DSM} est entraînée sur des tweets et non-tweets et utilise le schéma d'étiquette "BIOo". Il est important de noter que nous n'avons pas effectué une nouvelle recherche du meilleure paramètre [24] pour le système à base de dictionnaire.

Après l'extraction des EP prédites par le système LSR, le nouveau dictionnaire contient environ 14 991 EP uniques, soit une augmentation de plus 11000 EP qui n'étaient pas dans le dictionnaire de base. Nous récupérons toutes les EP annotées automatiquement par la configuration LSR_{ST-DSM}, c'est pour cela qu'il y a une si grande augmentation des EP.

5.8.2 Résultats obtenus

Ici, nous présentons les résultats de notre approche en deux étapes, utilisant la configuration LSR_{ST-DSM} et le dictionnaire. La table 5.8 contient les résultats obtenus pour les mesures d'évaluation *MWE-based*, *Token-based* et *MWE-link-based* sur l'ensemble de test de DiMSUM. Nous observons que l'utilisation séquentielle des systèmes LSR_{ST-DSM} et à base de dictionnaire surpasse le système LSR_{ST-DSM} sur les trois mesures d'évaluation. En effet, l'approche en deux étapes obtient 49,3% de score F1 *MWE-based*, soit environ 2,8 points de plus que le LSR_{ST-DSM}. Cependant, notre approche en deux étapes réduit la précision par rapport à la seule prédiction du système LSR_{ST-DSM}, et au contraire, notre approche augmente le rappel sur les données de tweets. Cela étant sûrement dû au fait que notre approche commet beaucoup d'insertions (environ 45% des erreurs) en reprenant les erreurs de prédictions du LSR_{ST-DSM} et au contraire permet d'annoter les EP correctement trouvées par le LSR_{ST-DSM} à chaque fois qu'elles se présentent dans une phrase. Une idée d'amélioration serait d'effectuer un filtrage à base de règles sur les EP prédites par le système neuronal afin de réduire les insertions d'EP incorrectes.

Configurations	**MWE-based**			**Token-based**	**MWE-link-based**
	Précision	**Rappel**	**score F1**	**score F1**	**score F1**
Dictionnaire	45,5	21,0	28,7	28,5	25,9
LSR_{ST-DSM}	**60,4** ($\pm 2,5$)	37,9 ($\pm 0,9$)	46,5 ($\pm 0,3$)	56,8 ($\pm 1,0$)	54,0 ($\pm 1,5$)
Deux étapes	50,4 ($\pm 1,4$)	**48,4** ($\pm 0,3$)	**49,3** ($\pm 0,6$)	**58,0** ($\pm 0,6$)	**56,0** ($\pm 0,9$)

TABLE 5.8 – Résultats d'identification d'EP sur les tweets de test de DiMSUM. Pour chaque résultat, le score moyen et l'écart type de 5 apprentissages sont donnés, sauf pour la configuration à base de dictionnaire (*Dictionnaire*). Les lignes *Dictionnaire* et LSR_{ST-DSM} sont recopiées de la table 5.4.

5.9 Conclusion du chapitre

Dans ce chapitre, nous avons étudié la robustesse de systèmes d'identification d'EP sur des tweets. Nous avons également proposé une approche en deux étapes.

Nous avons étudié deux types de systèmes d'identification d'EP existants : le premier basé sur un dictionnaire et le second sur des réseaux de neurones. Pour le système neuronal, nous avons proposé et évalué 7 configurations. Dans ces configurations, nous avons varié les jeux de données

24. Nombre de mots pouvant intervenir entre les UL d'une EP.

d'apprentissage et les schémas d'étiquettes. Nous avons constaté que toutes les configurations du système neuronal surpassent le système à base de dictionnaire. De plus, la meilleure configuration du système neuronal est celle qui est entraînée sur des tweets et des données non-tweets. Nous avons également observé que les étiquettes de super-sens et les catégories des EP n'aident pas pour l'identification des EP dans les tweets. La meilleure configuration du système neuronal utilise le schéma d'étiquetage "BIOo".

Nous avons proposé une nouvelle approche en deux étapes utilisant les deux types de systèmes. Le système neuronal prédit les EP dans le jeu de données cible, puis les EP prédites sont ajoutées au dictionnaire existant permettant au système à base de dictionnaire d'annoter le même jeu de données cible avec les EP découvertes préalablement par le système neuronal. Nous avons remarqué que cette approche surpasse l'utilisation du système neuronal.

6

Les expressions polylexicales pour la détection de la parole haineuse

Dans ce chapitre, nous présentons notre contribution pour la détection de la parole haineuse en proposant des approches automatiques intégrant les expressions polylexicales (EP). Notre objectif est d'étudier l'impact des EP (présentées dans le chapitre 5) sur la tâche de la détection de la parole haineuse dans les tweets. Pour cela, nous proposons deux architectures neuronales fondées sur des plongements de phrases (*sentence embeddings* en anglais) pour intégrer les EP. Pour chacun de nos systèmes neuronaux utilisant les EP, nous proposons des études de l'utilité des EP pour la détection des discours de haine.

Ce chapitre s'articule de la façon suivante : nous motivons et exposons nos questions de recherche dans la section 6.1. Dans la section 6.2, nous présentons l'intégration des informations des EP en entrée d'un système neuronal. Dans la section 6.3 nous décrivons nos configurations expérimentales. Pour finir, nous présentons les résultats obtenus (section 6.4) et nous concluons le chapitre (section 6.5).

Ce chapitre reprend et développe les résultats que nous avons présenté dans les conférences suivantes : *26th International Conference on Natural Language & Information Systems* (Zampieri et al., 2021) et *13th Language Resources and Evaluation Conference* (Zampieri et al., 2022b).

6.1 Motivations et questions de recherche

Pour rappel, les EP sont définies comme un groupe de lexèmes qui présentent une idiosyncrasie lexicale, morphologique, syntaxique, sémantique et/ou statistique (Baldwin and Kim, 2010). Stanković et al. (2020) ont montré que les EP aident à la détection de phrases haineuses pour la langue serbe. Les auteurs ont proposé d'ajouter les EP à caractère haineux dans un dictionnaire de termes haineux et d'utiliser ce dernier pour détecter les discours haineux. La construction d'un tel dictionnaire représente un travail complexe nécessitant l'intervention d'un expert natif de la langue cible afin de connaître les EP haineuses. Waseem et al. (2018) soutiennent que les EP sont importantes à prendre en compte pour la détection des discours de haine. Ils ont proposé de concaténer les mots d'une EP pour la représenter comme une seule unité lexicale (UL) (*take off* –> *take_off*). Cependant, ils ont uniquement appliqué ce traitement sur les EP continues composées de deux mots. Or, les mots d'une EP peuvent être contigus ou bien séparés par des mots intermédiaires qui n'appartiennent pas à cette expression. De plus, une EP peut être constituée de plus de deux UL. Il est important de noter qu'il n'existe aucune approche automatique utilisant les caractéristiques des EP pour la détection des discours de haine.

Ici, nous proposons des approches *automatiques* basées sur des réseaux neuronaux en utilisant les EP. Nous souhaitons répondre aux questions de recherche suivantes :

(1) Comment intégrer les EP dans un système neuronal ?

(2) L'intégration des EP dans un système neuronal est-elle utile pour la détection de la parole haineuse ?

(3) Quel système d'identification d'EP est le plus performant pour la détection de la parole haineuse ?

Pour intégrer les informations des EP dans un système neuronal et répondre à la question (1), nous proposons deux solutions qui nous permettent par la suite de répondre aux questions de recherche (2) et (3).

6.2 Méthodologie proposée

Dans cette section, nous établissons notre méthodologie pour l'intégration des EP pour la détection des discours de haine. Comme dans le chapitre 4, nous utilisons un système de base fondé sur les plongements de phrases. L'objectif est de l'améliorer en intégrant les EP. Nous proposons donc deux systèmes intégrant les EP de manières différentes : un système à deux branches neuronales (section 6.2.1) et un autre à trois branches (section 6.2.2). Ces systèmes sont inspirés du système neuronale proposé pour intégrer des caractéristiques linguistiques dans le chapitre 4.

6.2.1 Système neuronal à deux branches

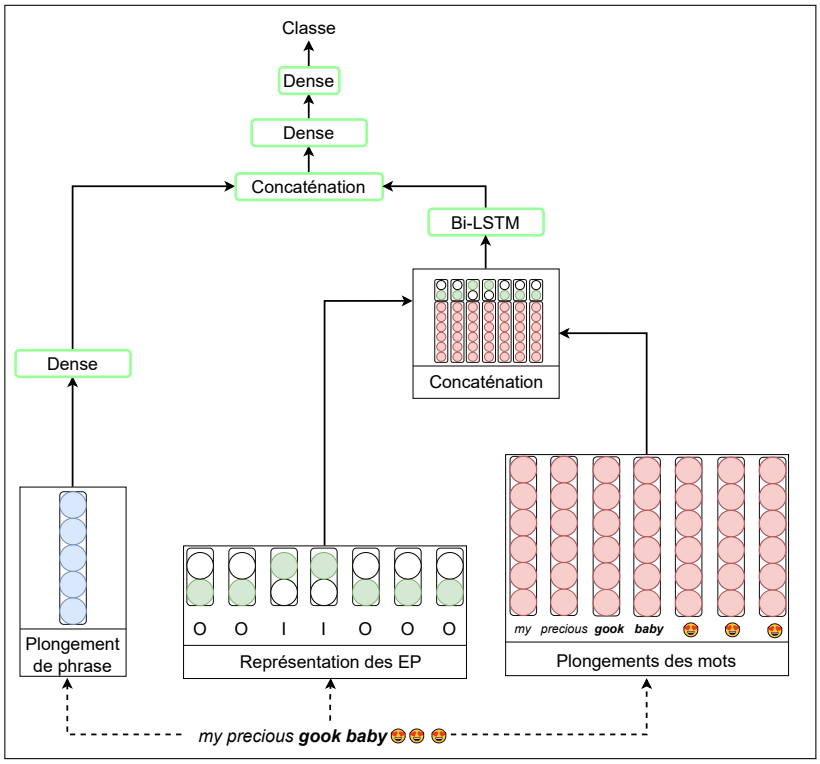


FIGURE 6.1 – Proposition d'un système neuronal basé sur les *embeddings* de phrase et sur les caractéristiques des EP concaténées aux *embeddings* des mots (DPH-2B).

Nous illustrons l'architecture neuronale du système de détection de la parole haineuse utilisant deux branches (DPH-2B) dans la figure 6.1.

Dans la première branche neuronale, le système DPH-2B utilise les plongements de phrases (en bleu dans la boîte *plongement de phrase* de la figure 6.1) qui sont donnés en entrée d'une couche dense. Cette couche permet de mettre des poids sur les plongements de phrases avant la concaténation avec la deuxième branche.

C'est dans la deuxième branche que nous intégrons les EP. L'idée est de mettre les EP dans leur contexte, c'est-à-dire d'utiliser les EP avec des plongements de mots qui sont issus générés d'un modèle de langage fondé sur les *transformers* car ces plongements de mots prennent en compte le contexte de la phrase. Nous concaténons (voir boîte *Concaténation* de la figure 6.1) les plongements des mots (en rouge dans la boîte *plongements des mots*) avec les étiquettes des EP (en vert dans la boîte *Etiquettes des EP*). Il est important de noter que tous les plongements des mots sont présents. La sortie de cette concaténation est donnée à une couche de Bi-LSTM pour apprendre des représentations des phrases en tenant compte des EP et des plongements de mots.

Les deux branches sont concaténées avant d'être données en entrée de deux couches denses pour la classification des tweets.

6.2.2 Système neuronal à trois branches

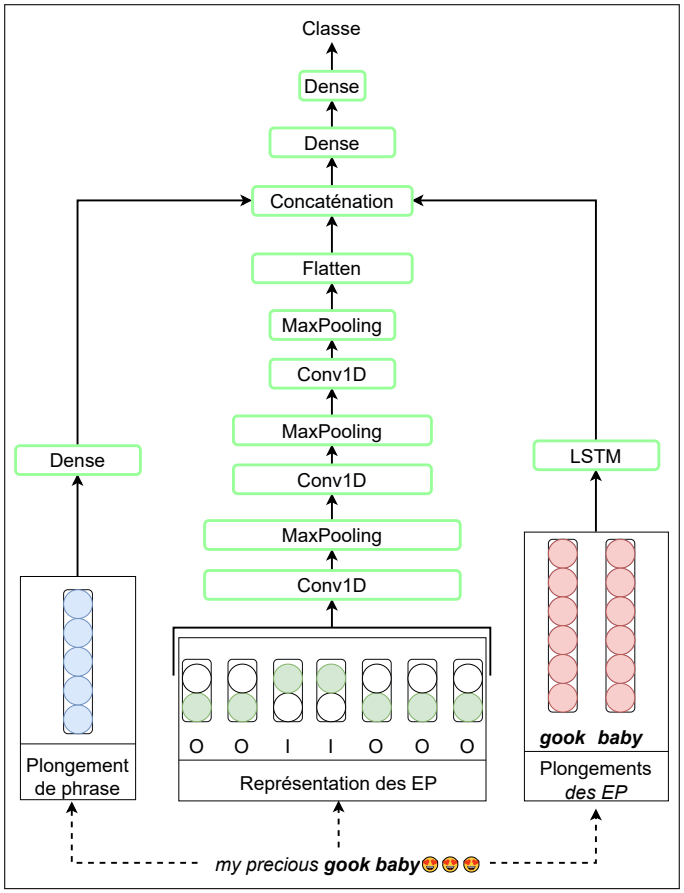


FIGURE 6.2 – Proposition d'un système neuronal basé sur les plongements de phrases et utilisant les étiquettes et plongements des EP (DPH-3B).

La figure 6.2 montre l'architecture neuronale du système de détection de la parole haineuse à trois branches (DPH-3B). La première branche neuronale (boîte *plongements de phrases* dans la figure 6.2) est identique à celle du système DPH-2B.

Concernant la deuxième branche neuronale (boîte *Etiquettes des EP* dans la figure 6.2), nous représentons les EP par un schéma d'étiquetage binaire qui sont données en entrée et passent par trois couches neuronales convolutionnelles à une dimension (Conv1D) avec une réduction de dimension (*maxpooling*), dont la sortie de la dernière couche de réduction de dimension est transformée en un vecteur (*flatten*), comme dans notre proposition de système dans le chapitre 4 (section 4.2.2).

Dans la troisième branche neuronale (boîte *plongements des EP* dans la figure 6.2), nous représentons les plongements des mots appartenant à une EP qui sont utilisés en entrée d'une couche de LSTM.

Comme pour le système DPH-2B, la sortie des branches neuronales est concaténées avant d'être données à deux couches denses successives.

6.3 Configurations expérimentales

Dans cette section, nous présentons les configurations des deux systèmes proposés.

6.3.1 Représentations des entrées des systèmes

Pour représenter les phrases entières, nous utilisons les plongements de phrases générés par l'encodeur USE comme dans le chapitre 4. Tous les systèmes (de base, DPH-2B et DPH-3B) utilisent ces plongements (en bleu dans les figures 6.1 et 6.2) dont la taille est de 512.

Pour représenter les étiquettes des EP (en vert dans les figures 6.1 et 6.2), nous utilisons un schéma d'étiquetage binaire : l'étiquette "I" signifie que le mot appartient à une EP et "O" le mot n'appartient à aucune EP. Nous avons testé différents schémas d'étiquetage, comme "BIO" et "BIOo", mais nous n'avons pas observé d'amélioration lors du développement des systèmes.

Concernant les plongements des mots (en rouge dans les figures 6.1 et 6.2), pour les systèmes DPH-2B et DPH-3B, nous extrayons les plongements du modèle de langage *BERTweet* (Nguyen et al., 2020). Pour rappel, BERTweet est pré-entraîné sur des tweets.

6.3.2 Annotation automatique des EP

Les jeux de données de la haine ne sont pas annotés en termes d'EP et nous utilisons les systèmes étudiés dans le chapitre précédent (chapitre 5) pour les annoter automatiquement en EP. Nous utilisons les systèmes d'identification d'EP suivants : basé sur un dictionnaire (voir section 5.4.1, basé sur des réseaux neuronaux (voir section 5.4.2) et combinant les deux (voir section 5.8). Pour chacun des systèmes nous utilisons les mêmes configurations que dans le chapitre précédant.

Concernant le système à base de dictionnaire, appelé *Dico* dans la suite du chapitre, nous utilisons le dictionnaire créé automatiquement en extrayant les EP des jeux de données annotés en termes d'EP (présenté en section 5.4.1). Afin de pouvoir utiliser ce système, nous effectuons automatiquement la lemmatisation des jeux de données de la haine en utilisant la librairie Python *spacy-udpipe*[25].

25. `https://spacy.io/universe/project/spacy-udpipe`

Pour le système neuronal, nous utilisons la configuration LSR_{ST-DSM} (décrite en section 5.5.2). Nous rappelons que la configuration LSR_{ST-DSM} est entraînée sur des tweets (ensemble d'apprentissage de DimSum) et des données d'autres médias d'internet (ensemble d'apprentissage de Streusle) avec un schéma d'étiquettes "BIOo". Comme nous avons effectué cinq entraînements avec une initialisation aléatoire, ici nous utilisons l'entraînement qui a permis d'obtenir le résultat médian sur le jeu de test pour l'identification des EP dans les tweets.

Concernant l'approche en deux étapes, nommés *LSR-Dico* dans la suite du chapitre, nous extrayons les EP de la prédiction du LSR_{ST-DSM} sur les ensembles d'apprentissage des jeux de données de la haine pour agrandir le dictionnaire existant de l'approche par dictionnaire. Ensuite, nous appliquons le système à base de dictionnaire avec ce nouveau dictionnaire pour étiqueter les EP dans les différents ensembles des jeux de données de la haine.

6.3.3 Paramètres des couches neuronales

Le système de base possède les mêmes paramètres que celui utilisé dans le chapitre 4 (section 4.3), à savoir que les deux couches de denses successives ont respectivement 256 et 1 (pour HatEval et Waseem) ou 3 (pour Davidson et Founta) neurones.

Pour le système DPH-2B, la couche dense de la première branche neuronale possède 256 neurones. Nous utilisons une taille de 256 (2x128) neurones pour la couche de Bi-LSTM. Les couches denses suivant la concaténation des deux branches neuronales possèdent le même nombre de neurones que celles du système de base.

Pour le système DPH-3B, nous avons choisi les mêmes paramètres que notre système proposé dans le chapitre 4 pour les couches Conv1D : la première est de taille 32, la deuxième de taille 16 et la dernière de taille 8. Pour rappel, La taille de fenêtre des couches Conv1D est de 3. Concernant la couche LSTM, elle est de taille 192. Ces paramètres ont été ajustés lors de la phase de développement. Les deux dernières couches denses du systèmes ont les mêmes paramètres que les systèmes de base et DPH-2B. Ce système possède environ un million de paramètres ajustables. Nous résumons les paramètres des différents systèmes dans la table 6.1.

Comme dans le chapitre précédent, nous utilisons un optimiseur de type *Adam* avec un *learning rate* de 1e-3. Comme les jeux de données Waseem et HatEval sont annotés en deux classes (non-haineux et haineux), nous utilisons une activation *sigmoid* pour la dernière couche dense avec une entropie croisée binaire pour le calcul d'erreur. Concernant les jeux de données Davidson et Founta, la dernière couche dense utilise une activation *softmax* et un calcul d'erreurs de type entropie croisée par catégories, car ces jeux de données répartissent les tweets dans trois classes.

Paramètres des systèmes			
DPH-3B		**DPH-2B**	
Couches	Tailles	Couches	Tailles
Conv1D	32 16 8	Bi-LSTM	256
LSTM	192		
Dense	256	Dense	256
Dense	#classe	Dense	#classe

TABLE 6.1 – Paramètres des systèmes DPH-2B et DPH-3B.

6.3.4 Apprentissages et mesures d'évaluation

Comme pour notre configuration expérimentale dans le chapitre 4, pour chaque système, nous procédons à cinq entraînements sur chacun des ensembles d'apprentissage avec une initialisation aléatoire. Chaque entraînement a 100 époques maximum avec un arrêt forcé (*early stopping* en anglais) basé sur le taux d'erreur minimal atteint sur l'ensemble de validation et une patience de cinq époques. Nous rappelons que les jeux de données pour la haine sont décrits dans le chapitre 3. Nous évaluons les différents systèmes sur les quatre jeux de données comme dans le chapitre 4. Nous utilisons comme mesure d'évaluation le score *macro-F1* (introduit dans le chapitre 3). Nous rappelons que pour déterminer s'il y a une amélioration significative par rapport au système de base, nous utilisons un test de paires appariées avec un risque de 5% (Gillick and Cox, 1989).

6.4 Résultats obtenus

La table 6.2 montre les scores *macro-F1* médians obtenus sur les différents ensembles de test. Dans un premier temps, nous répondons à notre première question de recherche : "intégrer les EP dans un système neuronal est-il utile pour la détection de la haine ?". Dans un second temps, nous répondons à notre deuxième question de recherche : "quel système d'identification d'EP est plus performant sur la détection de la parole haineuse ?". Pour finir, nous comparons nos deux propositions de systèmes intégrant les EP.

Systèmes DPH	**Systèmes id. d'EP**	**Classification binaire**		**Classification ternaire**		**Moyenne**
		Waseem	**HatEval**	**Davidson**	**Founta**	
Base	-	79,5 ($\pm 0,1$)	63,9 ($\pm 0,4$)	72,2 ($\pm 0,5$)	72,4 ($\pm 0,7$)	72,0
DPH-3B	Dictionnaire	81,3 ($\pm 0,2$)	**66,4** ($\pm 0,2$)	72,3 ($\pm 0,9$)	71,7 ($\pm 0,4$)	72,9
	LSR_{ST-DSM}	80,9 ($\pm 0,4$)	66,2 ($\pm 0,7$)	71,8 ($\pm 2,5$)	72,0 ($\pm 0,6$)	72,7
	Deux étapes	80,9 ($\pm 0,5$)	66,1 ($\pm 1,4$)	70,5 ($\pm 2,7$)	72,2 ($\pm 1,3$)	72,4
DPH-2B	Dictionnaire	**82,3** ($\pm 0,7$)	65,2 ($\pm 0,5$)	72,7 ($\pm 2,1$)	73,8 ($\pm 0,6$)	73,5
	LSR_{ST-DSM}	81,9 ($\pm 0,6$)	64,6 ($\pm 1,1$)	**73,8** ($\pm 1,4$)	**74,5** ($\pm 0,7$)	**73,7**
	Deux étapes	80,9 ($\pm 0,5$)	65,4 ($\pm 1,4$)	73,5 ($\pm 1,5$)	73,5 ($\pm 1,2$)	73,3

TABLE 6.2 – Médianes des scores *macro-F1* et écart-types sur 5 entraînements. Les résultats soulignés indiquent une amélioration significative par rapport au système de base. La colonne *Systèmes d'identification d'EP* représente les systèmes utilisés pour identifier les EP dans les jeux de données. La colonne *Moyenne* représente la moyenne des scores médians sur les quatre jeux de données et l'amélioration significative est calculée en concaténant toutes les prédictions. Les meilleurs résultats sont en gras pour chacun des ensembles de test.

6.4.1 Comparaison des systèmes DPH-2B et DPH-3B

Il est important de rappeler que la différence majeure entre les systèmes DPH-2B et DPH-3B est que le système DPH-3B utilise les plongements des mots appartenant à une EP et le système DPH-2B utilise tous les plongements des mots de phrase. Nous remarquons que le système DPH-2B obtient des scores *macro-F1* plus élevés comparé au système DPH-3B : environ 1% d'augmentation relative des scores *macro-F1* en moyenne. Cela est probablement dû au fait que le système DPH-2B utilise tout le contexte de la phrase en concaténant les étiquettes des EP aux plongements contextuels des mots de BERTweet, alors que le système DPH-3B utilise uniquement les plongements contextuels des mots appartenant à une EP.

6.4.2 Intégrer les EP dans un système neuronal est-il utile pour la détection de la haine ?

Pour le système DPH-3B, nous remarquons que l'intégration des EP améliorent les résultats sur les jeux de données Waseem et HatEval comparé au système de base. En effet, sur Waseem le système de base obtient un score de *macro-F1* de 79,5% et le système DPH-3B obtient un score de *macro-F1* de 81,3% et 80,5% avec respectivement les EP identifiées par les systèmes Dico et LSR_{ST-DSM}. Sur HatEval, nous constatons une amélioration significative de l'utilisation des EP étiquetées dans le système DPH-3B : 66,4% et 66,2% pour le système DPH-3B contre 63,9% pour le système de base. Sur Davidson et Founta, nous remarquons aucune amélioration significative des résultats avec le système DPH-3B comparé au système de base.

Concernant le système DPH-2B, nous observons des scores *macro-F1* supérieurs comparés à ceux obtenus par le système de base. En effet, le système DPH-2B améliore significativement les scores *macro-F1* comparés au système de base sur les jeux de données à classification ternaire (Davidson et Founta). Sur Davidson, le système de base obtient 72,2% de *macro-F1* contre 72,7% et 73,8% pour le système DPH-2B utilisant respectivement les EP prédites par les systèmes Dico et LSR_{ST-DSM}. Sur Founta, nous constatons que le système DPH-2B montre une amélioration relative de l'ordre de 1,9% à 2,9% du score *macro-F1* obtenu par le système de base. Nous observons également que sur les jeux de données Waseem et HatEval, le système DPH-2B obtient de meilleurs scores de *macro-F1* : sur Waseem 82,3% et 81,9% de score *macro-F1* pour le système DPH-2B comparé à 79,5% pour le système de base et sur HatEval 65,2% et 64,6% de score *macro-F1* pour le système DPH-2B contre 63,9% pour le système de base.

La table 6.2 montre que le système de base obtient un score de *macro-F1* de 72,0% en moyenne sur les quatre ensembles de test. Nous remarquons que les systèmes DPH-3B et DPH-2B atteignent un meilleur score de *macro-F1* moyen sur l'ensemble des jeux de données indépendamment du système d'identification d'EP utilisé comparé au système de base. Le système DPH-3B obtient des scores de 72,9% et 72,7% de *macro-F1* moyen avec les EP étiquetées par les systèmes Dico et LSR_{ST-DSM}. Le système DPH-2B surpasse le système de base en atteignant des scores *macro-F1* de 73,5% et 73,7% en utilisant respectivement les EP prédites par les systèmes Dico et LSR_{ST-DSM}. Nous remarquons également une amélioration des scores *macro-F1* moyens sur les quatre ensembles de test avec l'utilisation des EP étiquetées automatiquement par le système d'identification d'EP LSR-Dico.

Nous en concluons que l'intégration des EP étiquetées automatiquement dans un système neuronal aident à la détection des discours haineux dans les tweets.

6.4.3 Quel système d'identification d'EP est plus performant pour la détection de la parole haineuse ?

Ici, nous comparons l'utilisation des systèmes d'identification d'EP pour la détection de la parole haineuse. Nous comparons les prédictions du système d'identification d'EP à base de dictionnaire et à base de réseaux neuronaux LSR_{ST-DSM}.

Concernant le système DPH-3B, nous remarquons que les prédictions du système Dico obtiennent un score de *macro-F1* moyen de 72,9% sur les quatre ensembles de test. En comparaison, le système DPH-3B utilisant les prédictions des EP des LSR_{ST-DSM} et LSR-Dico atteignent un score de *macro-F1* respectif de 72,7% et 72,4%. Nous constatons que sur l'ensemble des jeux de données, les résultats sont similaires entre l'utilisation des EP prédites par le système d'identification des EP à base de dictionnaire, celui à base de réseaux de neurones et le système en deux étapes.

Concernant le système DPH-2B, nous observons que les résultats sont similaires (aucune différence significative sur les ensembles de test), comme avec le système DPH-3B, entre l'utilisation des prédictions d'EP provenant des différentes systèmes d'identification d'EP.

Nous en concluons que les EP aident à la détection de la parole haineuse en utilisant les prédictions des EP à partir d'un système à base de dictionnaire, à base de réseaux de neurones avec la configuration LSR_{ST-DSM} ou avec la combinaison de ces deux systèmes (système en deux étapes).

6.4.4 Études d'ablation

Dans cette section, nous présentons une étude d'ablation pour chacun des systèmes proposés intégrant les EP : DPH-2B et DPH-3B. Pour ces études d'ablation nous utilisons les EP étiquetées automatiquement par les systèmes d'identification d'EP à base de dictionnaire et de réseaux neuronaux car c'est avec ces EP que les systèmes obtiennent les meilleurs scores *macro-F1* en moyenne.

Système DPH-2B

Pour l'étude d'ablation du système DPH-2B, nous comparons le système DPH-2B avec et sans l'utilisation de la concaténation des étiquettes des EP aux plongements des mots de la phrase (sans le cadre *Étiquettes des EP* de la figure 6.1). C'est-à-dire que nous étudions l'utilité des EP dans notre système DPH-2B. Dans la table 6.3 nous présentons les résultats obtenus avec et sans l'utilisation des étiquettes d'EP dans le système DPH-2B.

Systèmes DPH	**Systèmes id. d'EP**	**Classification binaire**		**Classification ternaire**		**Moyenne**
		Waseem	**HatEval**	**Davidson**	**Founta**	
Base	-	79,5 ($\pm 0,1$)	63,9 ($\pm 0,4$)	72,2 ($\pm 0,5$)	72,4 ($\pm 0,7$)	72,0
DPH-2B	Sans Etiq.	81,3 ($\pm 0,7$)	64,4 ($\pm 2,1$)	71,8 ($\pm 2,6$)	72,8 ($\pm 1,3$)	72,6
	Dictionnaire	**82,3** ($\pm 0,7$)	**65,2** ($\pm 0,5$)	72,7 ($\pm 2,1$)	73,8 ($\pm 0,6$)	73,5
	LSR_{ST-DSM}	81,9 ($\pm 0,6$)	64,6 ($\pm 1,1$)	**73,8** ($\pm 1,4$)	**74,5** ($\pm 0,7$)	**73,7**

TABLE 6.3 – Médianes des scores *macro-F1* et écart-types sur 5 entraînements des systèmes de base et DPH-2B, avec et sans EP. La colonne *Moyenne* représente la moyenne des scores médians sur les quatre jeux de données. *Sans Etiq.* signifie que le système n'utilise pas les EP. Les résultats soulignés indiquent une amélioration significative par rapport au système de base. Les meilleurs résultats sont en gras.

Nous constatons que le système DPH-2B n'utilisant pas les EP obtient un score de *macro-F1* de 72,6% en moyenne. Nous observons que le score *macro-F1* obtenu sans l'utilisation des EP est significativement inférieur aux scores de ce même système utilisant les étiquettes des EP : 73,5% et 73,7%. Cela confirme le fait que les EP sont utiles pour la détection de la parole haineuse. Nous constatons aussi que le contexte de la phrase, avec les plongements des mots en plus du plongement de la phrase, apporte une aide supplémentaire à la détection des discours haineux. En effet, nous remarquons une légère augmentation du score *macro-F1* moyen sur les quatre ensembles de test : 72,0% obtenu par le système de base (utilisant seulement les plongements de phrases) comparé à 72,6% pour le système DPH-2B n'utilisant aucune information sur les EP. Cependant cette amélioration des résultats n'est pas significative.

Système DPH-3B

Ici, nous effectuons l'analyse par ablation du système DPH-3B. Nous comparons l'utilité des étiquettes d'EP et des plongements des EP. Pour estimer l'importance des étiquettes des EP, nous supprimons la branche avec cette caractéristique en entrée (boîte *Etiquettes des EP* dans la figure 6.2). Pour analyser l'utilité des plongements des EP, nous supprimons cette information du système DPH-3B (boîte *Plongements des EP* dans la figure 6.2). Nous focalisons notre étude d'ablation du système DPH-3B avec l'utilisation des EP prédites par le système d'identification d'EP Dico, car c'est avec ces EP que le système a obtenu les meilleurs résultats en moyenne. La table 6.4 présente les résultats obtenus du système DPH-3B.

Systèmes DPH	**Caract. des EP**	**Classification binaire**		**Classification ternaire**		**Moyenne**
		Waseem	**HatEval**	**Davidson**	**Founta**	
Base	-	79,5 ($\pm 0,1$)	63,9 ($\pm 0,4$)	72,2 ($\pm 0,5$)	72,4 ($\pm 0,7$)	72,0
DPH-3B	Etiq.	81,1 ($\pm 0,4$)	66,0 ($\pm 0,4$)	70,1 ($\pm 1,6$)	71,6 ($\pm 0,7$)	72,2
	Plg.	81,0 ($\pm 0,3$)	65,4 ($\pm 0,5$)	67,6 ($\pm 1,6$)	70,9 ($\pm 0,8$)	71,2
	Etiq. et Plg.	**81,3** ($\pm 0,2$)	**66,4** ($\pm 0,2$)	**72,3** ($\pm 0,9$)	**71,7** ($\pm 0,4$)	**72,9**

TABLE 6.4 – Médianes des scores *macro-F1* et écart-types sur 5 entraînements des systèmes de base et DPH-3B. La colonne *Caract. des EP* représente les caractéristiques des EP utilisées : *Etiq.* et *Plg.* signifient respectivement que le système DPH-3B utilise les étiquettes des EP et/ou les plongements des EP. La colonne *Moyenne* représente la moyenne des scores médians sur les quatre jeux de données. Les résultats soulignés montrent une amélioration significative comparé au système de base. Les meilleurs résultats sont en gras.

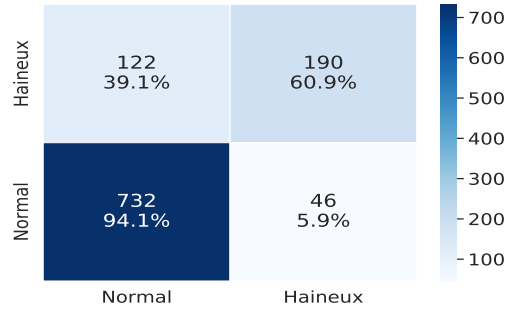
Nous remarquons qu'utiliser uniquement les étiquettes des EP améliore légèrement le score *macro-F1* moyenne sur les quatre ensembles de données : 72,2% contre 72,0% pour le système de base. Nous observons qu'utiliser uniquement les plongements des mots appartenant aux EP dégrade les résultats : 71,2% de score *macro-F1* en moyenne. Il est important de noter que cette dégradation est observée sur tous les jeux de données avec trois classes. En effet, le système DPH-3B utilisant que les plongements des mots appartenant aux EP obtient 67,6% et 70,9% de score *macro-F1* sur respectivement Davidson et Founta. Pour rappel, le système de base atteint 72,2% et 72,4% sur les ensembles de test de Davidson et Founta.

Nous constatons que le système DPH-3B, utilisant à la fois les étiquettes des EP et les plongements des mots appartenant aux EP obtient les meilleurs scores *macro-F1* sur tous les ensembles de test comparé à l'utilisation séparément de l'une des deux caractéristiques.

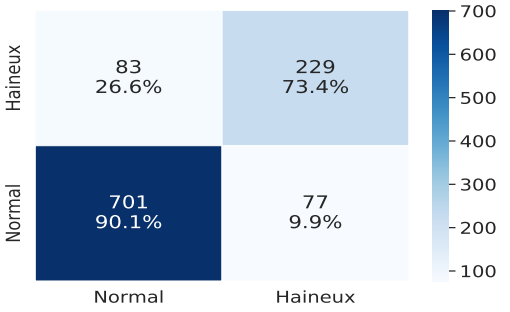
6.4.5 Matrices de confusion

Dans cette section, nous analysons les matrices de confusion sur les quatre ensembles de test. Nous comparons les matrices de confusion du système de base, qui n'utilise pas les caractéristiques des EP et le système DPH-2B utilisant les EP générées par le système d'identification d'EP LSR_{ST-DSM} car ce système a obtenu les meilleurs scores pour la détection des discours de haine. La figure 6.3 montre les différentes matrices de confusion pour chacun des jeux de données.

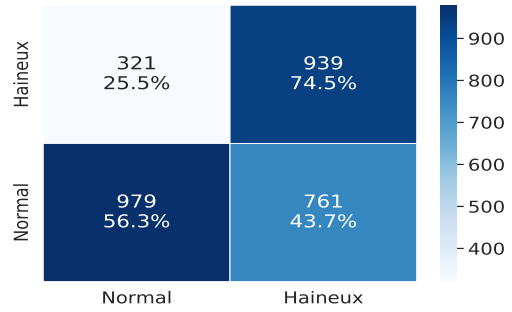
Concernant les matrices de confusion sur Waseem (figure 6.3(a) et 6.3(b)), nous constatons que l'utilisation des EP améliore la classification des tweets haineux. En effet, le système DPH-2B classifie correctement 73,4% des tweets haineux comparé au système de base qui en détecte correctement 60,9%. En contrepartie, le système DPH-2B commet plus d'erreurs que le système de base sur la détection des tweets normaux.



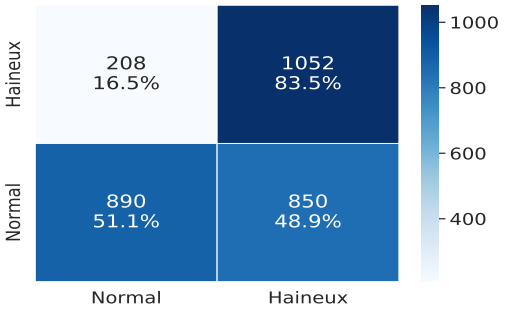
(a) Système de base : Waseem



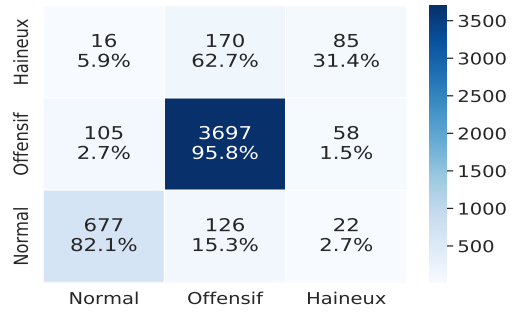
(b) DPH-2B : Waseem



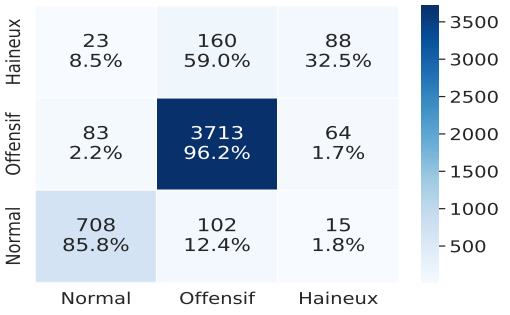
(c) Système de base : HatEval



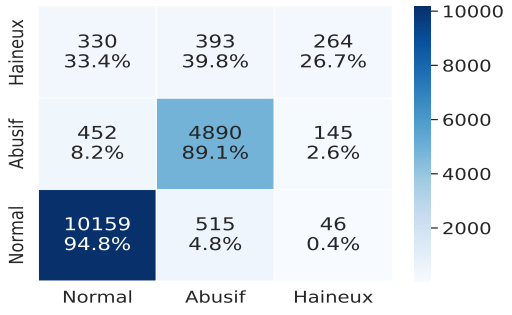
(d) DPH-2B : HatEval



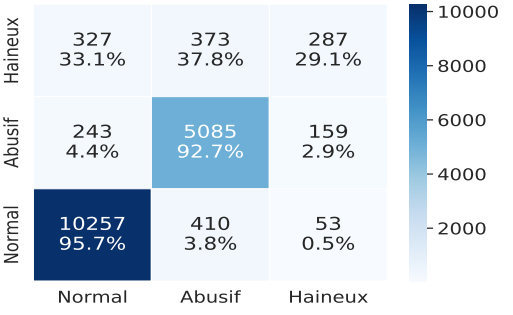
(e) Système de base : Davidson



(f) DPH-2B : Davidson



(g) Système de base : Founta



(h) DPH-2B : Founta

FIGURE 6.3 – Matrices de confusion sur tous les ensembles de test pour les systèmes de base (a, c, e, g) et DPH-2B utilisant les étiquettes des EP générées par le système d'identification d'EP LSR_{ST-DSM} (b, d, h, g).

En effet, nous remarquons une légère dégradation de la détection des tweets normaux : 9,9% de tweets normaux mal détectés par le système DPH-2B comparé à 5,9% pour le système de base. Cependant, la dégradation est moins importante comparé à l'amélioration apportée par le système DPH-2B : 3,9% de dégradation de la détection des tweets normaux comparé à 12,5% d'amélioration de la détection des tweets haineux.

En comparant les matrices de confusion du système de base (figure 6.3(c)) et DPH-2B (figure 6.3(d)) sur l'ensemble de test de HatEval, nous remarquons que l'amélioration est similaire à celle obtenue sur le jeu de données Waseem. En effet, le système DPH-2B détecte mieux les tweets haineux (8,9% d'amélioration) au détriment de la classification des tweets normaux (5,1% de dégradation). Comme sur Waseem, nous constatons que l'amélioration de la détection des tweets haineux est supérieur à la dégradation observée sur les tweets normaux.

Nous illustrons respectivement les matrices de confusion sur l'ensemble de test de Davidson pour le système de base et DPH-2B dans les figures 6.3(e) et 6.3(f). Nous observons que sur cet ensemble de test, l'utilisation des étiquettes des EP avec le système DPH-2B améliore la classification des tweets normaux, offensifs et haineux. En effet, le système de base détecte correctement 82,1%, 95,8% et 31,4% des tweets normaux, offensifs et haineux comparé à 85,8%, 96,2% et 32,5% correctement identifié par le système DPH-2B, soit une amélioration de 3,7%, 0,4% et 1,1% pour respectivement la détection des tweets normaux, offensifs et haineux avec le système DPH-2B.

L'amélioration de la détection des tweets haineux du système DPH-2B est due à la correction des tweets classifiés comme offensifs par le système de base. Concernant la classification des tweets offensifs et normaux, nous remarquons que le système DPH-2B corrige les tweets identifiés comme normaux au lieu d'offensifs par le système de base et inversement les tweets détectés comme offensifs par le système de base alors qu'ils sont de la classe *normal*.

Concernant les matrices de confusion sur l'ensemble de test illustrées figures 6.3(g) et 6.3(h), nous observons le même phénomène que sur le jeu de données Davidson. Le système DPH-2B améliore la classification des tweets normaux, abusifs et haineux. En effet, le système de base détecte correctement 94,8%, 89,1% et 26,7% des tweets normaux, abusifs et haineux comparé à 95,7%, 92,7% et 29,1% correctement identifié par le système DPH-2B, soit une amélioration de 0,9%, 3,6% et 2,4% pour respectivement la détection des tweets normaux, abusifs et haineux avec le système DPH-2B.

Concernant la détection des tweets haineux, nous remarquons que cette amélioration est due à une réduction de 0,3% et 2,0% (relatif) de l'erreur de classification des tweets haineux identifiés comme étant respectivement normaux et abusifs par le système de base. Nous observons que le système DPH-2B corrige les tweets identifiés comme *normal* au lieu de *abusif* : 8,2% de tweets offensifs détectés comme normaux par le système de base comparé à 4,4% pour le système DPH-2B, soit une réduction de moitié de cette erreur. Concernant la légère amélioration de la détection des tweets normaux, nous constatons que le système DPH-2B corrige les tweets normaux identifiés comme abusifs par le système de base.

6.4.6 Analyse de quelques exemples

Dans cette section, nous analysons quelques exemples des classifications des différents systèmes. La table 6.5 présente les tweets avec les classes attendues et prédites par les différents systèmes de détection des discours haineux sur deux tweets : "*Suck shit Kat you dumb shit . Go back into the swamp behind ya and shut the fuck up .*" et "*Brunette Lexy Little has lesbian sex experience of her lifetime with Lioness*".

Exemples (sources)	**Classe attendue**	**Systèmes**				
		de base	**DPH-3B**		**DPH-2B**	
			Dico	**LSR**	**Dico**	**LSR**
Suck shit *Kat you* ***dumb shit*** *. Go back into the swamp behind ya and* ***shut*** *the* ***fuck up*** *.* (Waseem)	Haine	Normal	Normal	Haine	Haine	Haine
Brunette Lexy Little *has* ***lesbian sex experience*** *of her lifetime with Lioness* (Founta)	Abusif	Normal	Abusif	Abusif	Abusif	Abusif

TABLE 6.5 – Exemples issus des ensembles de test avec les classes de haine attendues et prédites par les différents systèmes de détection des discours haineux. Les mots soulignés représentent les EP détectées par le système à base de dictionnaire. Les mots en gras représentent les EP identifiées par le système LSR_{ST-DSM}. *Dico* et *LSR* font respectivement référence à l'utilisation des EP prédites par les systèmes d'identification d'EP à base de dictionnaire et à base de réseaux neuronaux LSR_{ST-DSM}.

Concernant le premier tweet de la table 6.5 ("*Suck shit Kat you dumb shit . Go back into the swamp behind ya and shut the fuck up .*"), les EP identifiées automatiquement sont : *Go back* et *fuck up* par le système Dico et *Shut shit*, *dumb shit* et *shut fuck up* par le système LSR_{ST-DSM}. Nous constatons que les systèmes de détection de la haine de base et DPH-3B utilisant les EP prédites par le système Dico prédisent la mauvaise classe : classe *normal*. A l'inverse, les systèmes DPH-3B utilisant les EP prédites par le système LSR_{ST-DSM} et DPH-2B (indépendamment des systèmes d'identification d'EP utilisés) prédisent la bonne classe pour ce tweet : classe *haine*. Concernant le système DPH-3B utilisant les EP prédites par le système LSR_{ST-DSM}, cela est probablement dû au fait que les EP prédites sont fortement haineuses (*Shut shit*, *dumb shit* et *shut fuck up*). Concernant le système DPH-2B, il parvient à bien classifier ce tweet indépendamment des systèmes d'identification d'EP utilisés. Cela est probablement dû au fait qu'il contient dans tous les cas les plongements des mots de la phrase et parvient à extraire le contexte haineux qui n'est peut-être pas contenu dans le plongement de la phrase.

Concernant le deuxième tweet de la table 6.5 ("*Brunette Lexy Little has lesbian sex experience of her lifetime with Lioness*"), les EP identifiées automatiquement sont : *has experience* par le système Dico et *Brunette Lexy Little* et *lesbian sex experience* par le système LSR_{ST-DSM}. Nous constatons que les systèmes DPH-3B et DPH-2B (indépendamment des systèmes d'identification d'EP utilisés) classent correctement le tweet comme *abusif*, alors que le système de base classe le tweet comme *normal*. Concernant le système DPH-2B, la bonne classification de ce tweet est probablement due aux faits que toutes les EP contenant *Brunette*, *sex* et *lesbian* apparaissent dans plus de 90% des cas dans des tweets abusifs de l'ensemble d'apprentissage de Founta. De plus, le contexte des plongements de tous les mots de la phrase peuvent aider aussi le système. Pour le système DPH-3B utilisant les EP prédites par le dictionnaire, la bonne classification de ce tweet est probablement due aux plongements de l'EP *has experience* qui sont contextuels et donc peuvent prendre en comptes les autres mots de la phrase. Concernant la bonne classification de ce tweet avec les EP étiquetées par le système LSR_{ST-DSM} cela est probablement due aux mêmes faits que la bonne classification du système DPH-2B.

6.5 Conclusion du chapitre

Dans ce chapitre, nous avons présenté notre contribution de l'étude de l'intégration des EP pour la détection de la parole haineuse. Nous avons répondu à nos deux questions de recherche

qui étaient : "est-ce qu'intégrer les EP dans un système neuronal aide à la détection de la haine ?", "quel système d'identification d'EP est plus performant pour notre tâche : à base de dictionnaire ou à base de réseaux neuronaux ?" Pour cela, nous avons proposé deux systèmes neuronaux intégrant les EP : DPH-2B et DPH-3B.

Les systèmes DPH-2B et DPH-3B utilisent les EP en tant que caractéristiques d'entrée. Pour les deux systèmes, nous avons constaté que les caractéristiques des EP aident à la détection de la parole haineuse sur quatre jeux de données de la haine. Nous avons également remarqué une amélioration de la détection des discours de haine avec les trois systèmes d'identification d'EP utilisées. Nous avons constaté que les meilleurs résultats sont obtenus par le système DPH-2B avec les étiquettes d'EP prédites par le système d'identification d'EP LSR_{ST-DSM}. De plus, pour chacun des systèmes intégrant les EP, nous avons réalisé une étude par ablation afin de montrer l'importance des EP pour notre tâche. Nous en concluons que les EP, utilisées comme caractéristiques d'entrée, sont importantes pour la détection des discours de haine.

7

Approche multi-tâches avec mécanisme d'auto-attention pour la détection des discours haineux

Dans ce chapitre, nous étudions l'utilisation de l'apprentissage multi-tâches pour la détection des discours de haine. Pour cela nous utilisons comme tâche auxiliaire à la détection de la parole haineuse l'identification des expressions polylexicales (EP). Dans le chapitre précédent, nous avons montré que les EP sont utiles pour classifier les discours de haine. Notre intuition est qu'un système multi-tâches apprendra des caractéristiques importantes pour l'identification des EP, ce qui aidera la classification des contenus haineux. Nous proposons un système utilisant un mécanisme d'attention afin d'apprendre les deux tâches simultanément.

Ce chapitre est organisé de la façon suivante : nous motivons notre recherche (section 7.1) et nous exposons notre méthodologie (section 7.2). Ensuite nous détaillons les configurations expérimentales utilisées (section 7.3). Enfin, nous présentons les résultats obtenus pour la détection de la parole haineuse (section 7.4).

Ce chapitre reprend et approfondi les résultats publiés dans la conférence *10th Language & Technology Conference* (Zampieri et al., 2023).

7.1 Motivations

L'apprentissage multi-tâches permet d'apprendre simultanément des caractéristiques provenant de plusieurs tâches. Des chercheurs ont montré l'intérêt de l'utilisation de cette méthode pour la détection de la parole haineuse en utilisant des tâches étroitement liées aux discours de haine, tels que l'analyse de sentiments (Rajamanickam et al., 2020; Zhou et al., 2021; Chiril et al., 2022; Plaza-Del-Arco et al., 2021; del Arco et al., 2021), l'identification des cibles des discours de haine (Awal et al., 2021a), la classification discours de haine en utilisant plusieurs jeux de données (D'Sa et al., 2022; Kapil and Ekbal, 2020), ou bien la reconnaissance d'entités nommées (Montariol et al., 2022).

Dans ce chapitre, nous nous intéressons à l'apprentissage simultané de deux tâches : l'identification d'EP et la détection des discours de haine. Dans le chapitre précédent nous avons montré que les EP, utilisées comme caractéristiques d'entrée de systèmes neuronaux, aidaient à la détection des discours de haine. Notre hypothèse est que l'utilisation d'un tel apprentissage permettra d'apprendre des caractéristiques de la tâche d'identification d'EP pour servir la tâche de détection des discours de haine. Pour cela, nous proposons d'utiliser un mécanisme d'auto-

attention permettant de généraliser sur ces deux tâches. C'est pour cela que nous nous posons les questions de recherche suivantes :

— Est-il important d'incorporer un mécanisme d'attention pour apprendre à faire attention aux EP et aux discours de haine simultanément ?

— L'apprentissage multi-tâches pour la détection des discours de haine est-il efficace avec pour tâche secondaire l'identification des EP ?

Pour répondre à ces questions nous proposons un système neuronale utilisant un mécanisme d'attention et le paradigme de l'apprentissage multi-tâches.

7.2 Méthodologie proposée

Nous proposons un système de détection de la haine multi-tâches avec un mécanisme d'auto-attention. Les deux tâches sont la détection des discours haineux et l'identification des EP.

7.2.1 Mécanisme d'attention

Le mécanisme d'attention a été introduit dans le domaine du TAL avec la tâche de traduction automatique afin de pallier au problème de la perte du contexte qu'ont les réseaux de neurones récurrents (RNN) sur des longues phrases (Bahdanau et al., 2014). L'idée est que le système fasse attention à tous les mots en entrée pour prédire la séquence traduite. Pour cela, les auteurs proposent d'utiliser les couches cachés d'un encoder pour apprendre une matrice d'attention. Yin et al. (2016) ont étudié différents mécanismes d'attention sur différentes tâches du TAL, tels que la détection d'implication textuelle, l'identification de paraphrase, etc. Lin et al. (2017) ont proposé un mécanisme d'attention pour obtenir des représentations de mots plus sensibles au contexte dans la phrase, connu sous le nom d'auto-attention.

Nous souhaitons que le système apprenne automatiquement à faire attention aux EP pour améliorer la détection de la parole haineuse. Chakrabarty et al. (2019) ont montré que l'attention contextuelle (Yang et al., 2016) améliore plus les résultats que l'auto-attention à une seule tête. Zhou et al. (2021) ont aussi utilisé un mécanisme d'auto-attention multi-têtes pour leur approche multi-tâches. De plus, le mécanisme d'auto-attention multi-têtes a montré être de grande performance pour apprendre des représentations générales dans le TAL. C'est le cas avec les modèles de langages fondés sur les *transformers* tels que BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019a), etc.

Dans notre cas, nous implémentons un mécanisme d'auto-attention multi-têtes. L'intuition est que certaines têtes d'attention se focalisent sur la première tâche du système, tandis que les autres portent attention à la seconde tâche. La figure 7.1 montre l'auto-attention par produit scalaire (*scaled dot-product* en anglais) multi-têtes utilisé dans notre système. La formule de l'auto-attention est la suivante :

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

avec QK^T correspondant à la matrice de dépendance entre les mots de la phrase. Q, K, V représentent respectivement les vecteurs de requête (*query* en anglais), de clé (*key* en anglais) et de valeur (*value* en anglais). $\sqrt{d_k}$ représente la dimension du vecteur de clé.

Pour chaque mot dans une phrase donnée, trois vecteurs sont créés : un vecteur de clé, un vecteur de valeur et un vecteur de requête. Ces vecteurs sont calculés en multipliant la matrice de plongements des mots (*word embeddings* en anglais) de la phrase par des matrices de poids

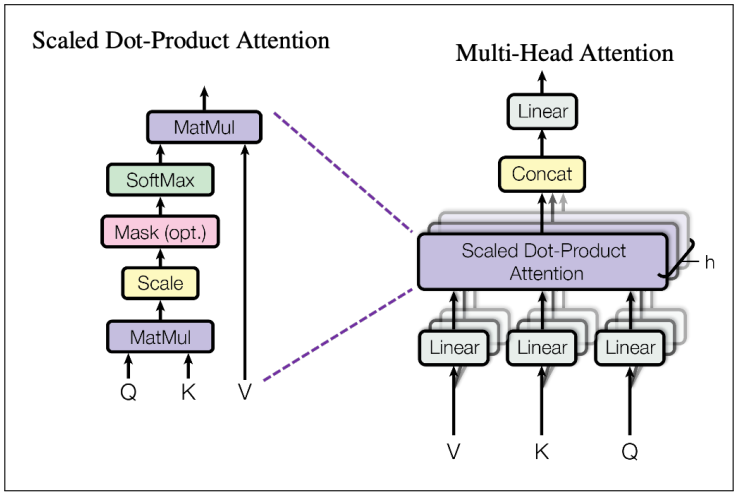


FIGURE 7.1 – Auto-attention à plusieurs têtes utilisant le produit scalaire. Figure extraite de l'article *All you need is attention* (Vaswani et al., 2017a).

spécifiques. Pour chaque paire de mots, un score d'attention est calculé en effectuant un produit scalaire entre le vecteur de clé et le vecteur de requête. Ces scores indiquent l'importance de chaque mot par rapport aux autres dans le contexte de la phrase. Les scores d'attention sont ensuite normalisés à l'aide de la fonction *softmax*. Cela permet d'obtenir des poids normalisés indiquant la distribution de l'attention entre les mots. Les vecteurs de valeurs sont ensuite pondérés par les poids normalisés d'attention. Ces vecteurs pondérés sont ensuite sommés pour obtenir le vecteur de sortie finale pour chaque mot. Ce vecteur de sortie contient des informations contextuelles provenant de tous les autres mots de la phrase.

7.2.2 Système multi-tâches avec mécanisme d'auto-attention

La figure 7.2 montre l'architecture proposée de notre système multi-tâches avec mécanisme d'auto-attention à plusieurs têtes.

En entrée du système nous donnons les mots de la phrase à classifier. Les mots sont représentés par des plongements. Ces derniers sont mis en entrée d'une couche d'auto-attention dont la sortie est utilisée pour deux tâches : prédire les EP de la phrase et détecter la classe de haine de cette dernière. Pour identifier les EP (boîte verte de la figure 7.2), le système utilise une couche neuronale dense distribuée dans le temps (*time distributed* en anglais) afin de prédire une étiquette pour chacun des mots de la phrase. Concernant la détection des discours haineux (boîte rouge de la figure 7.2), le système utilise une couche de Bi-LSTM, afin de calculer une représentation de la phrase, suivie d'une couche dense pour la classification de la phrase en terme de haine. La couche d'auto-attention apprend à faire attention, en simultanée, aux EP et à la haine. Pour l'affinage du modèle, la fonction de perte suivante est calculé :

$$\mathcal{L} = \lambda \mathcal{L}_{DPH} + \beta \mathcal{L}_{EP} \tag{7.1}$$

où λ et β sont deux paramètres qui respectivement donnent de l'importance à la fonction de perte pour la détection de la haine ($\mathcal{L}_{DPH}$) et l'identification des EP ($\mathcal{L}_{EP}$).

Pour entraîner un tel système, deux solutions sont possibles :

— utiliser un jeu de données spécifique pour la détection des discours de haine et un autre spécifique pour l'identification d'EP ;

— utiliser un jeu de données annoté en termes de haine et d'EP.

Concernant la première solution le système apprendra à résoudre une tâche après l'autre. De plus, nous avons remarqué que les jeux de données annotés en termes d'EP contiennent peu de phrases haineuses et donc cela augmenterait le nombre de messages normaux observés par le système lors de l'entraînement. Donc, nous optons pour la deuxième solution. Il est important de rappeler qu'il n'existe pas de jeux de données avec une telle annotation et annotés manuellement un jeu de données est long et coûteux. Donc nous utilisons une méthode automatique. Il n'est pas envisageable d'annoter en termes de haine le seul jeu de données de tweets étiqueté en EP (DiMSUM), car il contient moins de 1000 tweets d'apprentissage. De plus, nous avons remarqué que très peu de tweets de DiMSUM étaient haineux (moins de 1% des tweets). Donc, nous annotons automatiquement en termes d'EP les jeux de données pour la détection des discours de haine. Nous utilisons la prédiction des EP effectuée par le système neuronal LSR_{ST-DSM}, car nous avons montré dans le chapitre précédent (chapitre 6) que c'est avec ce système d'identification d'EP que les meilleurs résultats sont obtenus.

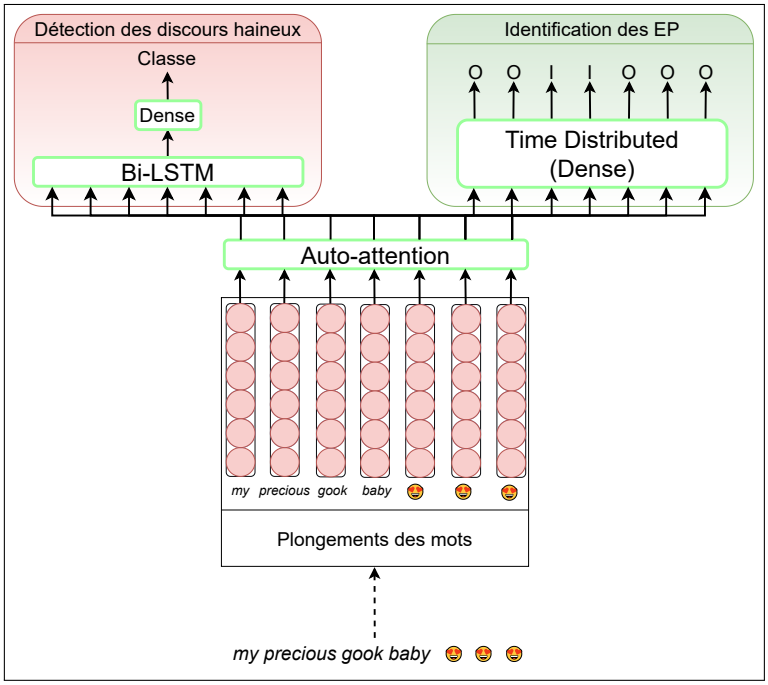


FIGURE 7.2 – Architecture de notre système multi-tâches avec un mécanisme d'auto-attention. Les deux tâches sont la détection des discours haineux et l'identification des EP.

7.3 Configurations expérimentales

Ici, nous décrivons les configurations expérimentales pour notre système multi-tâches avec mécanisme d'auto-attention.

7.3.1 Représentations des mots

Nous utilisons plusieurs plongements de mots générés à partir de modèles de langage état de l'art : BERTweet (Nguyen et al., 2020), HateBERT (Caselli et al., 2021) et fBERT (Sarkar et al., 2021). Nous utilisons ces trois modèles de langage car ils sont entraînés sur différents jeux de

données. Le modèle BERTweet est entraîné sur des millions de tweets. HateBERT est entraîné sur des commentaires potentiellement haineux de Reddit et les auteurs de ce modèle ont montré qu'il obtenait des performances impressionnantes sur notre tâche (Caselli et al., 2021). Le modèle fBERT est un modèle BERT ré-entraîné sur des tweets offensifs. Les créateurs de ce modèle ont montré qu'il surpassait HateBERT pour la détection des discours de haine (Sarkar et al., 2021). Tous ces modèles de langage génèrent des plongements de mots de taille 768.

7.3.2 Configurations des couches neuronales

Nous varions le nombre de têtes d'attention : nous utilisons 2 et 4 têtes d'attention. Concernant les dernières couches, la couche Bi-LSTM possède 256 neurones (2x128) et la couche dense a la même configuration que les classifieurs neuronaux utilisés dans les chapitres précédents. Comme pour nos précédentes expériences pour les jeux de données possédant une tâche de classification binaire (Waseem et HatEval) nous utilisons une activation *sigmoid* et pour les jeux de données à classification ternaire (Davidson et Founta) nous utilisons une activation *softmax*. Nous avons fixé λ et β a 0,5 pour le calcul de la fonction de perte du système multi-tâches (équation 7.1). Ces paramètres sont issus d'une recherche d'hyperparamètres lors du développement du système.

Concernant la dernière couche dense pour l'identification des EP, nous utilisons une activation *softmax* avec deux d'étiquettes à prédire dû au schéma d'étiquetage "IO" utilisé. Dans le schéma d'étiquetage "IO", les mots appartenant à une EP sont étiquetés par "I", les autres mots sont étiquetés avec "O". Lors du développement de notre système multi-tâches, nous avons testé différents schémas d'étiquetage, tels que "BIO", et "BIOo" (introduits dans le chapitre 5), mais nous avons remarqué qu'utiliser un jeu d'étiquettes binaire obtenait les meilleurs performances pour la détection des discours de haine. Cela est probablement dû au fait que les EP sont annotés automatiquement dans les jeux de données pour la détection des discours de haine. Nous pensons qu'avec une annotation manuelle des EP dans les jeux de données, d'autres schémas d'étiquetage aurait pu être étudiés pour notre tâche.

7.3.3 Apprentissages et mesure d'évaluation

Chaque entraînement a 100 époques maximum avec un arrêt forcé (*early stopping* en anglais) basé sur le taux d'erreur minimal atteint sur l'ensemble de validation et une patience de cinq époques. Nous évaluons les différents systèmes sur les quatre jeux de données comme dans les chapitres 4 et 6. Nous rappelons que pour déterminer s'il y a une amélioration significative par rapport au système de base, nous utilisons un test de paires appariées avec un risque de 5% (Gillick and Cox, 1989).

7.4 Résultats obtenus

Tout d'abord, nous comparons les performances des différents plongements utilisés pour représenter les mots des tweets Pour rappel, nous comparons les plongements de mots de BERTweet, HateBERT et fBERT. Dans la table 7.1, nous observons que les plongements de BERTweet atteignent un score de *macro-F1* de 72,2% en moyenne sur tous les ensembles de test. En comparaison, le système avec les plongements de HateBERT et de fBERT obtiennent respectivement des scores *macro-F1* de 72,7% et de 72,3% en moyenne. Nous remarquons que l'utilisation des plongements de mots de fBERT et HateBERT obtient de légèrement meilleurs score moyenne comparé à BERTweet, et cela est probablement dû au fait que ces modèles de langage ont vu lors

de leurs apprentissages des contenus à caractères haineux et offensifs. Pour rappel, HateBERT est entraîné sur des messages de Reddit potentiellement haineux et fBERT est basé sur un modèle BERT ré-entraîné sur des tweets offensifs. Cependant, cette augmentation des scores *macro-F1* n'est pas significative et n'est pas représentative sur tous les jeux de données.

Systèmes DPH	**#Tête d'att.**	**Classification binaire**		**Classification ternaire**		**Moyenne**
		Waseem	**HatEval**	**Davidson**	**Founta**	
plongements BERTweet						
Tâche unique	-	82,8 (±0,5)	61,1 (±4,8)	71,0 (±0,6)	73,7 (±1,0)	72,2
	2	84,5 (±0,8)	61,5 (±2,3)	72,0 (±3,2)	74,0 (±0,9)	73,0
	4	84,3 (±2,0)	64,4 (±3,7)	73,2 (±2,3)	74,1 (±1,1)	74,0
multi-tâches	2	**85,8** (±2,2)	64,2 (±1,8)	74,2 (±1,0)	73,5 (±0,4)	**74,5**
	4	85,1 (±0,7)	63,3 (±4,6)	73,6 (±2,2)	74,1 (±1,1)	74,0
plongements HateBERT						
Tâche unique	-	81,6 (±1,2)	63,1 (±3,6)	71,9 (±3,5)	74,3 (±0,6)	72,7
	2	82,4 (±0,6)	61,0 (±2,4)	72,8 (±3,1)	73,8 (±1,7)	72,5
	4	82,7 (±1,6)	60,4 (±2,9)	73,2 (±1,8)	74,4 (±0,7)	72,7
multi-tâches	2	83,2 (±0,8)	63,7 (±2,6)	**75,1** (±2,0)	74,6 (±0,3)	74,2
	4	83,5 (±2,3)	**65,0** (±1,7)	73,3 (±3,1)	73,4 (±1,2)	73,8
plongements fBERT						
Tâche unique	-	80,7 (±2,1)	61,6 (±1,8)	71,9 (±2,0)	75,3 (±0,5)	72,3
	2	82,6 (±1,3)	56,6 (±3,3)	67,9 (±3,5)	74,7 (±1,2)	70,5
	4	84,2 (±3,2)	59,7 (±5,5)	73,2 (±1,8)	74,7 (±0,8)	73,4
multi-tâches	2	84,2 (±0,6)	58,7 (±4,1)	72,1 (±2,4)	74,5 (±1,2)	72,3
	4	83,9 (±1,4)	62,0 (±3,2)	68,6 (±3,8)	**75,6** (±0,7)	72,5

TABLE 7.1 – Médianes des scores *macro-F1* et écart-types sur 5 entraînements. La colonne *#Tête d'att.* représente le nombre de têtes pour la couche d'attention. Les résultats significativement meilleurs que le système de base (tâche unique sans auto-attention) sont soulignés. Les meilleurs résultats pour chacun des ensembles de test sont en gras.

7.4.1 Impact de l'auto-attention à plusieurs têtes

Ici, nous étudions l'impact de l'auto-attention à plusieurs têtes, sans utilisation de l'apprentissage multi-tâches. Nous avons présenté les résultats dans les figures 7.3,7.4 et 7.5.

Avec l'utilisation des plongements générés par BERTweet, nous constatons, dans la figure 7.3, qu'utiliser le mécanisme d'auto-attention à plusieurs têtes améliore les score *macro-F1* en moyenne comparé au système de base (bar rouge dans la figure 7.3) : 73,0% et 74,0% avec respectivement 2 et 4 têtes d'attention comparé à 72,2% de *macro-F1* obtenu par le système de base. De plus, nous remarquons que l'amélioration concernant l'utilisation de 4 têtes d'attention est significative comparé au système de base.

Concernant l'utilisation des plongements de HateBERT (voir figure 7.4), nous observons aucune amélioration des résultats en moyenne sur tous les ensembles de test, malgré une amélioration significative sur les ensembles de test de Davidson (73,2% contre 71,9%) et Founta (74,4% contre 74,3%) avec 4 têtes d'attention et sur l'ensemble de test de Davidson avec 2 têtes d'attention (72,8% contre 71,9%). Cela est notamment dû au fait de la dégradation des scores *macro-F1* sur le jeu de données HatEval : le système de base obtient 63,1% contre 61,0% et 60,4% respectivement avec l'utilisation de 2 et 4 têtes d'attention. Nous avons observé que cette

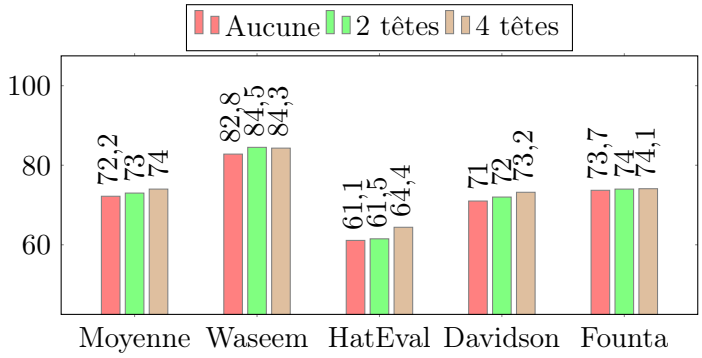


FIGURE 7.3 – Scores *macro-F1* des systèmes entraînés sur une tâche unique avec aucune, 2 et 4 têtes d'auto-attention et les plongements de BERTweet. Les résultats sont extraits de la table 7.1.

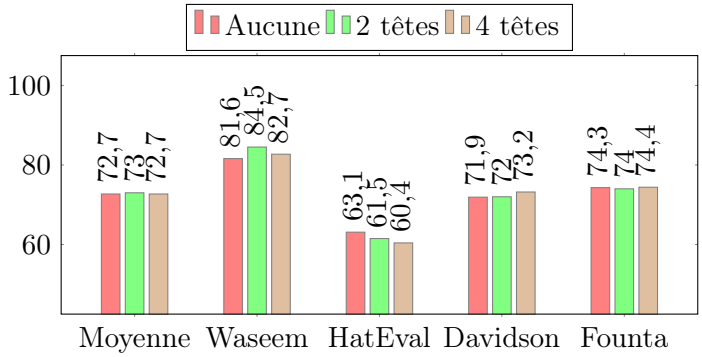


FIGURE 7.4 – Scores *macro-F1* des systèmes entraînés sur une tâche unique avec aucune, 2 et 4 têtes d'auto-attention et les plongements de HateBERT. Les résultats sont tirés de la table 7.1.

dégradation sur HatEval est due au fait qu'il prédit comme haineux trop de tweets normaux.

Concernant l'utilisation des plongements générés par fBERT (voir figure 7.5), nous constatons qu'utiliser 2 têtes d'attention dégrade les performances comparé au système de base : 70,5% de score *macro-F1* en moyenne sur tous les ensembles de test comparé à 72,3% de score *macro-F1* obtenu par le système de base. A l'inverse, le système utilisant 4 têtes d'attention montre une amélioration en moyenne sur les quatre jeux de données comparé au système de base : 73,4% contre 72,3%. Cette amélioration est notamment visible sur les ensembles de test de Waseem (84,2% contre 80,7%) et de Davidson (73,2% contre 71,9%). Cependant, ces améliorations ne sont pas significatives.

Pour résumer, nous constatons que le mécanisme d'auto-attention à plusieurs têtes montre une amélioration significative des résultats avec l'utilisation des plongements générés par BERTweet (notamment avec 4 têtes d'attention). A l'inverse, avec les plongements générés par HateBERT et fBERT, nous constatons aucune amélioration significative en moyenne sur tous les jeux de données.

7.4.2 Impact de l'apprentissage multi-tâches

Ici, nous présentons les résultats pour l'étude de l'impact de l'apprentissage multi-tâches simultanément sur la détection des discours haineux et sur l'identification des EP. Nous illustrons les résultats pour cette étude dans les figures 7.6, 7.7 et 7.8.

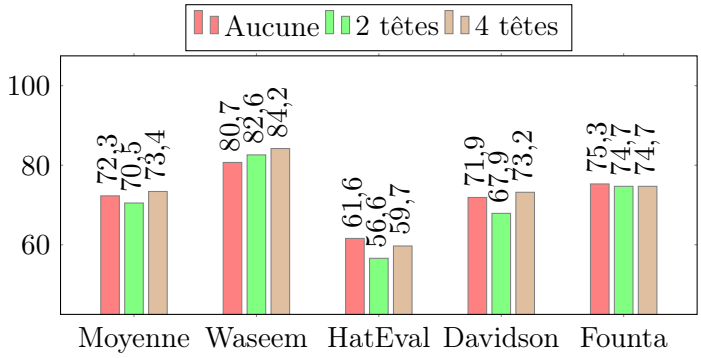


FIGURE 7.5 – Scores *macro-F1* des systèmes entraînés sur une tâche unique avec aucune, 2 et 4 têtes d'auto-attention et les plongements générés fBERT. Les résultats sont extraits de la table 7.1.

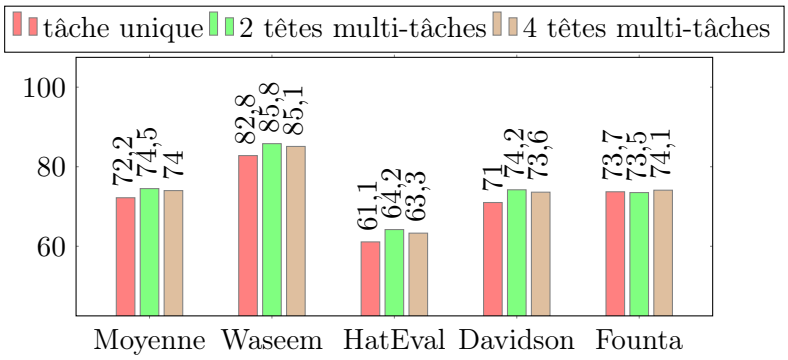


FIGURE 7.6 – Scores *macro-F1* du système à tâche unique sans auto-attention et du système multi-tâches utilisant 2 et 4 têtes d'auto-attention. Les plongements utilisés sont générés par BERTweet. Les résultats sont tirés de la table 7.1.

Nous observons, dans la figure 7.6, que le système multi-tâches avec auto-attention à plusieurs têtes surpasse le système de base utilisant une seule tâche sans auto-attention avec l'utilisation des plongements générés par BERTweet. Nous constatons une amélioration significative du système multi-tâches avec 2 et 4 têtes d'attention comparé au système de base : 74,5% et 74,0% de score *macro-F1* en moyenne avec respectivement 2 et 4 têtes d'attention contre 72,2% pour le système de base.

Dans la figure 7.7, en utilisant les plongements de mots HateBERT nous remarquons la même amélioration qu'avec les plongements de mots générés par BERTweet. Notre approche multi-tâches (74,2% et 73,8% avec respectivement 2 et 4 têtes d'attention) surpasse le système de base (72,7%). De plus, nous notons que les meilleurs scores *macro-F1* sont obtenus par le système multi-tâches avec 2 têtes d'attention pour les plongements générés par BERTweet et HateBERT.

Néanmoins, concernant l'utilisation des plongements provenant de fBERT (voir figure 7.8), nous notons que le système multi-tâches avec auto-attention obtient des résultats similaires au système de base utilisant ces mêmes plongements de mots : 72,3% et 72,5% de *macro-F1* avec 2 et 4 têtes d'attention contre 72,3% pour le système de base. Cela peut être dû au fait que le modèle de langue fBERT est très largement entraîné sur des données qui ne sont pas des tweets et qui ne sont pas potentiellement haineuses.

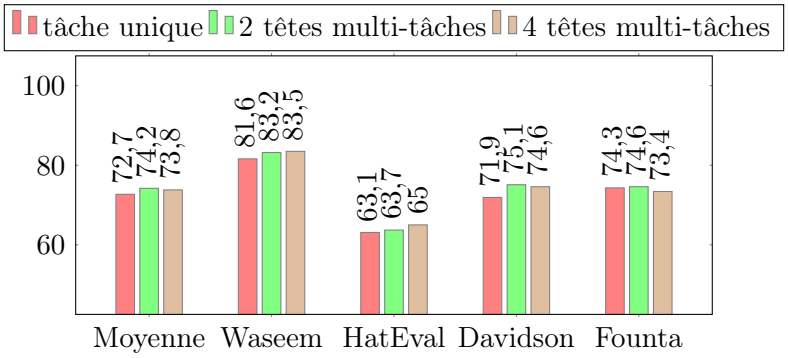


FIGURE 7.7 – Scores *macro-F1* du système à tâche unique sans auto-attention et du système multi-tâches utilisant 2 et 4 têtes d'auto-attention. Les plongements utilisés sont générés par HateBERT. Les résultats sont tirés de la table 7.1.

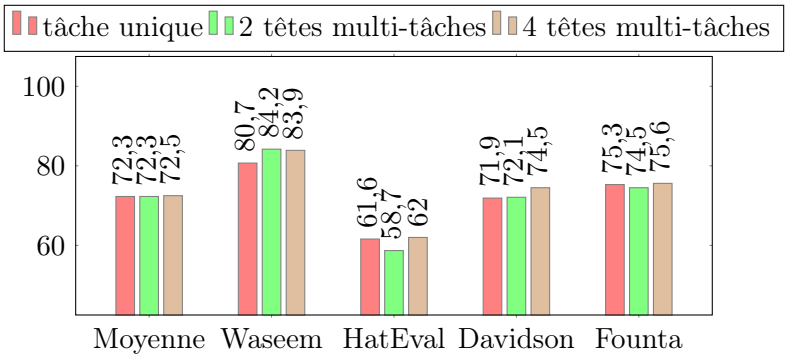


FIGURE 7.8 – Scores *macro-F1* du système à tâche unique sans auto-attention et du système multi-tâches utilisant 2 et 4 têtes d'auto-attention. Les plongements utilisés sont générés par fBERT. Les résultats sont extraits de la table 7.1.

7.4.3 Comparaison du système intégrant les EP avec le système multi-tâches à deux têtes d'auto-attention

Ici, nous comparons nos deux approches proposées utilisant les EP : la première utilise les EP en entrée d'un système neuronal (système DPH-2B décrit dans le chapitre 6) et la deuxième approche est notre système multi-tâches avec auto-attention multi-têtes qui consiste à faire attention simultanément aux caractéristiques des EP et des discours haineux. Pour rappel, le système DPH-2B utilise des plongements de phrase à classifier et de mots constituant cette phrase auxquels sont concaténés les étiquettes des EP au format "IO" (voir section 6.2.1). Afin de comparer les deux approches, nous utilisons le système DPH-2B avec les EP étiquetées à partir du système d'identification d'EP LSR_{ST-DSM} qui a obtenu le meilleur score de *macro-F1* moyen et qui est aussi utilisé pour apprendre la tâche d'identification d'EP de notre système multi-tâches. De plus, nous utilisons uniquement le système multi-tâches entraîné avec les plongements de BERTweet car ils sont aussi utilisés par le système DPH-2B.

Dans la figure 7.9, nous illustrons les scores *macro-F1* médians obtenus par les deux systèmes. Le système DPH-2B obtient un score de *macro-F1* de 73,7% en moyenne sur les quatre jeux de données. En comparaison, notre système multi-tâches avec auto-attention atteint un score de *macro-F1* de 74,5% et 74,0% avec respectivement 2 et 4 têtes d'attention. Nous remarquons une

augmentation du score *macro-F1* entre 0,3% et 0,8% avec l'apprentissage multi-tâches comparé au système DPH-2B. Donc il semble plus intéressant d'apprendre à faire attention aux EP avec le système multi-tâches à plusieurs têtes d'attention qui obtient de meilleurs résultats que les systèmes présentés dans le chapitre précédent (chapitre 6) : DPH-2B et DPH-3B. Néanmoins, il est important de noter que cette amélioration n'est pas significative.

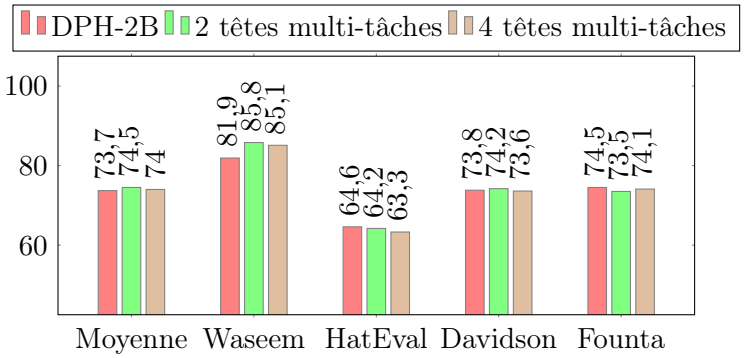


FIGURE 7.9 – Scores *macro-F1* du système DPH-2B et du système multi-tâches utilisant 2 et 4 têtes d'attention. Les plongements des mots sont générés par BERTweet. Les résultats sont reportés des tables 6.2 et .7.1

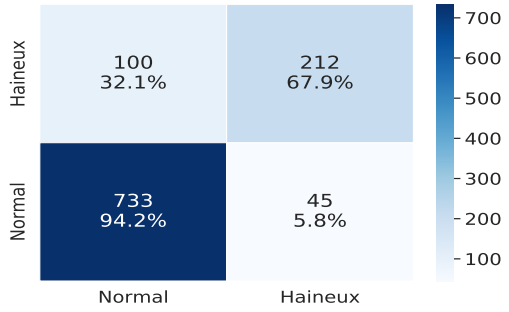
7.4.4 Matrices de confusion

Dans cette section, nous analysons les matrices de confusion du système de base et multi-tâches à deux têtes d'auto-attention. Pour les deux systèmes nous illustrons les matrices de confusion dans la figure 7.10 avec l'utilisation des plongements de mots générés par le modèle BERTweet.

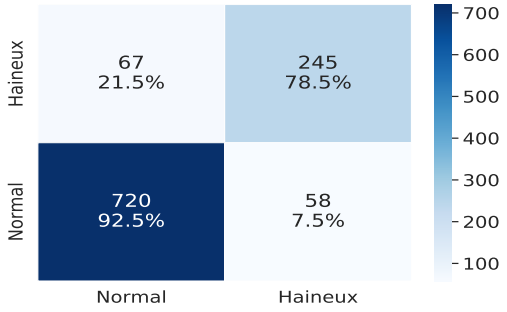
Concernant les matrices de confusion sur Waseem (voir figures 7.10(a) et 7.10(b)), pour les systèmes de base et multi-tâches, nous constatons que le système multi-tâches détecte mieux les tweets haineux comparé au système de base. Notre système détecte correctement 78,5% des tweets haineux comparé au système de base (67,9%). Cependant, le système proposé commet plus d'erreurs que le système de base de la détection des tweets normaux : 7,5% versus 5,8%. La dégradation est moins importante comparé à l'amélioration apportée par le système multi-tâches avec deux têtes d'auto-attention : 1,8% de dégradation relative sur l'identification des tweets normaux contre 15,6% d'amélioration relative de la détection des tweets haineux.

En comparant les matrices de confusion du système de base (figure 7.10(c)) et multi-tâches avec deux têtes d'auto-attention (figure 7.10(d)) sur l'ensemble de test de HatEval, nous observons le phénomène inverse que celui observé sur Waseem : le système multi-tâches montre une amélioration sur la détection des tweets normaux et une dégradation de la classification des tweets haineux. Cependant, le gain de la détection des tweets normaux (13,9% relatif) est largement supérieur à la dégradation de la classification des tweets haineux (1,8% relatif).

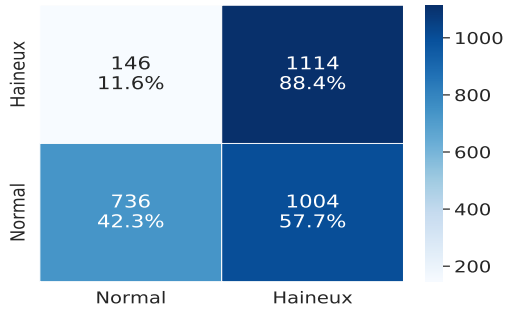
Pour l'ensemble de test de Davidson, nous illustrons respectivement les matrices de confusion des systèmes de base et multi-tâches dans les figures 7.10(e) et 7.10(f). Nous observons que, sur cet ensemble de test l'approche multi-tâches avec mécanisme d'auto-attention améliore la détection des tweets normaux et haineux au détriment de la classification des tweets offensifs. Nous remarquons que cette amélioration est dû à la correction des tweets classifiés comme offensifs par le système de base. En revanche, quelques tweets offensifs correctement classifiés par le système de base ont été identifiés comme normaux ou haineux par le système multi-tâches.



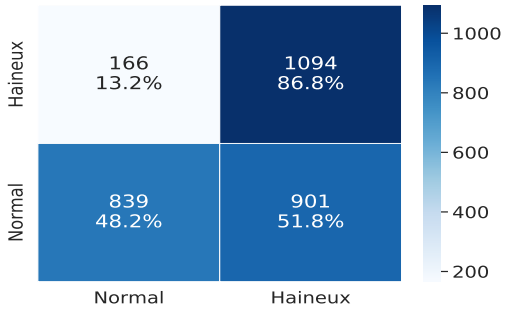
(a) Système de base : Waseem



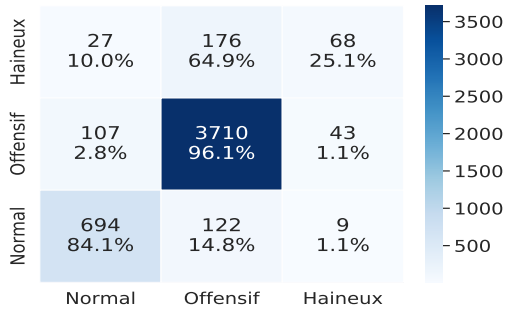
(b) Système multi-tâches : Waseem



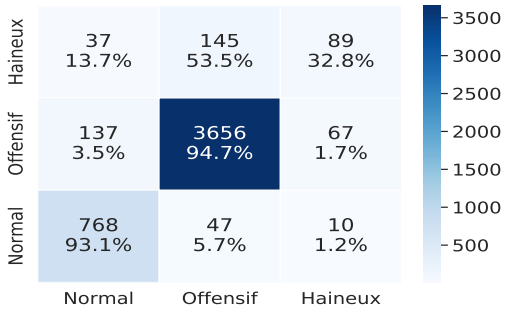
(c) Système de base : HatEval



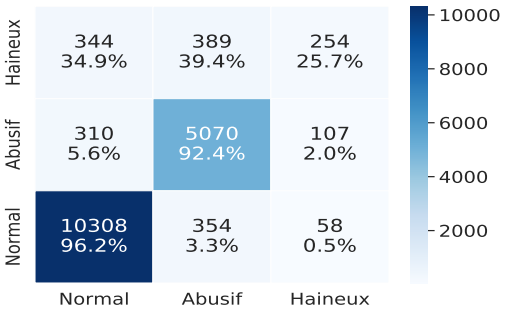
(d) Système multi-tâches : HatEval



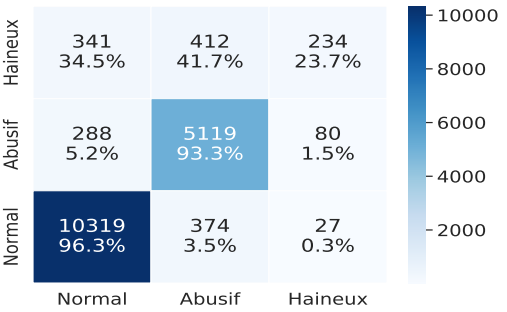
(e) Système de base : Davidson



(f) Système multi-tâches : Davidson



(g) Système de base : Founta



(h) Système multi-tâches : Founta

FIGURE 7.10 – Matrices de confusion sur tous les ensembles de test pour les systèmes de base (a, c, e, g) et multi-tâches avec 2 têtes d'auto-attention (b, d, f, h). Les deux systèmes utilisent les plongements de mots générés par BERTweet.

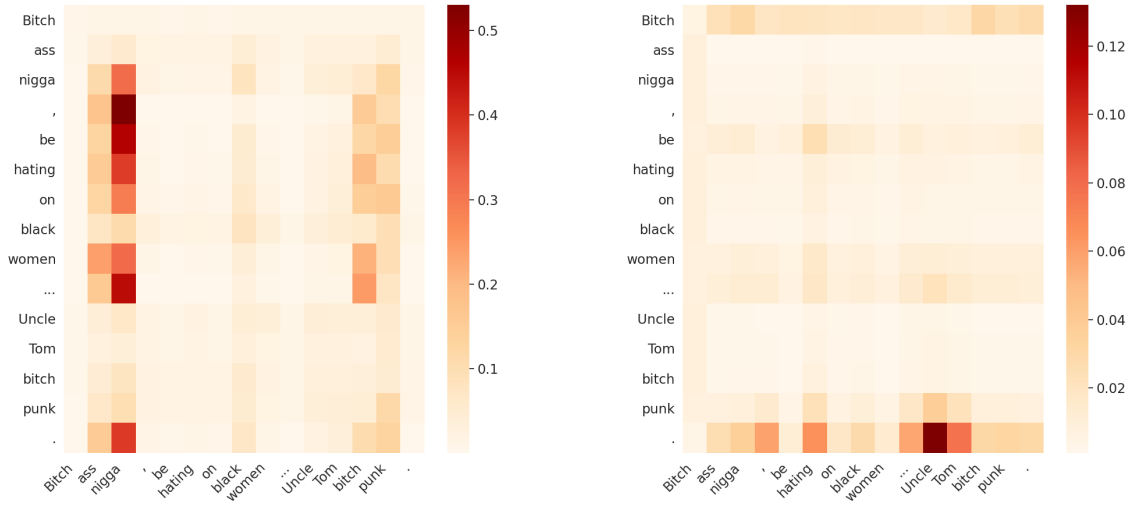
Cependant, nous remarquons que l'amélioration du système multi-tâches est supérieur comparé à la dégradation de la classification des tweets offensifs : 10,7% et 30,6% d'améliorations relatives de la détection respective des tweets normaux et haineux contre 1,5% de dégradation relative.

Concernant les matrices de confusion sur l'ensemble de test de Founta, illustrées respectivement dans les figures 7.10(g) et 7.10(h), nous observons que le système multi-tâches améliore légèrement la détection des tweets normaux et abusifs et dégrade légèrement la détection des tweets haineux par rapport au système de base. Contrairement aux observations sur les autres ensembles de test, nous ne remarquons pas de différences significatives des prédictions du système de base et multi-tâches.

7.4.5 Analyse des têtes d'attention

Dans cette section, nous proposons d'analyser les poids des têtes d'attention pour le système multi-tâches avec deux têtes d'attention utilisant les plongements générés par BERTweet. Pour rappel, une de nos hypothèse est que les différentes têtes d'attention se focalisent chacune sur une tâche précise.

La figure 7.11 montre les poids des deux têtes d'attention sur l'exemple haineux "*Bitch as nigga, be hating on black women... Uncle Tom bitch punk*" de l'ensemble de validation de Davidson. Nous remarquons que la première tête d'attention (figure 7.11(a)) porte l'attention sur les termes à caractères haineux : *ass*, *nigga* et *bitch*. Nous observons que la deuxième tête d'attention (figure 7.11(b)) porte plus d'attention à l'EP *Uncle Tom*. Nous constatons qu'une tête semble faire plus attention aux termes haineux et que la deuxième tête d'attention tend plus à faire attention aux EP.

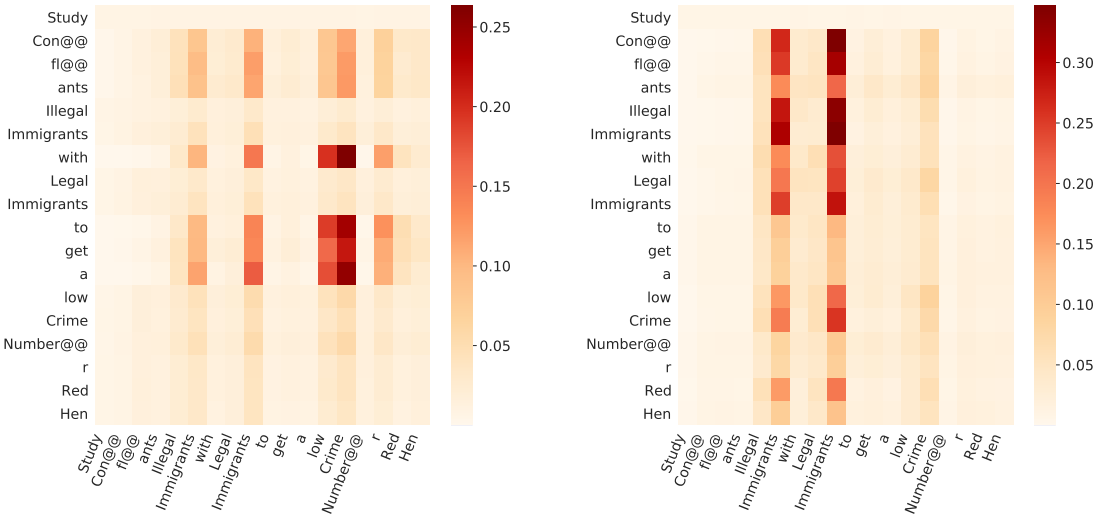


(a) Poids de la première tête d'attention

(b) Poids de la deuxième tête d'attention

FIGURE 7.11 – Poids d'attention du système multi-tâches avec deux têtes d'attention (a et b) utilisant les plongements de BERTweet. L'exemple haineux est issu de l'ensemble de validation de Davidson : "*Bitch as nigga, be hating on black women... Uncle Tom bitch punk*". L'exemple contient deux EP identifiées automatiquement par le système d'identification d'EP LSR_{ST-DSM} : *Bitch ass nigga* et *Uncle Tom*.

Dans la figure 7.12, nous exposons des matrices similaires sur un exemple haineux de l'ensemble de validation de HatEval : "*Study Conflants Illegal Immigrants with Legal Immigrants to get a low Crime Numberr Red Hen*". Dans cet exemple le système d'identification d'EP LSR_{ST-DSM} a détecté automatiquement deux EP : *Illegal Immigrants* et *Crime Numberr Red Hen*. Nous observons que la première tête d'auto-attention (figure 7.12(a)) se focalise plus sur les groupes de mots *Illegal Immigrants* et *low Crime Numberr*. Nous remarquons que la deuxième tête d'auto-attention montre une attention majoritairement orientée sur les deux occurrences du mot *Immigrants* qui est souvent utilisé dans les discours haineux. Ces observations viennent conforter notre hypothèse sur l'utilisation de plusieurs têtes d'attention.



(a) Poids de la première tête d'attention

(b) Poids de la deuxième tête d'attention

FIGURE 7.12 – Poids d'attention du système multi-tâches avec deux têtes d'attention (a et b) utilisant les plongements de BERTweet. L'exemple haineux est issu de l'ensemble de validation de HatEval : "*Study Conflants Illegal Immigrants with Legal Immigrants to get a low Crime Numberr Red Hen*". L'exemple contient deux EP identifiées automatiquement par le système d'identification d'EP LSR_{ST-DSM} : *Illegal Immigrants* et *Crime Numberr Red Hen*.

7.5 Conclusion du chapitre

Dans ce chapitre, nous avons présenté notre contribution à l'étude de l'apprentissage multi-tâches qui avait pour principal objectif de répondre à la question de recherche : "*est-il important d'incorporer un mécanisme d'attention pour apprendre à faire attention aux EP et aux discours de haine simultanément ?*". Pour cela, nous avons utilisé les tâches d'identification des EP et de détection des discours de haine. Nous avons proposé un système neuronal avec un mécanisme d'auto-attention multi-têtes fondé sur des plongements de mots.

Pour cette étude, nous avons évalué différents plongements de mots : générés par BERTweet, HateBERT et fBERT. Nous avons observé que ces différents plongements de mots obtenaient des performances similaires dans le système de base. Cependant, nous avons remarqué qu'avec les plongements de mots de BERTweet et de HateBERT, notre système multi-tâches obtenait les meilleurs performances. Nous avons montré par ablation que notre système multi-tâches avec attention a surpassé le système de base. En comparant notre approche avec un système intégrant

les EP en tant que caractéristique, nous avons observé que le système multi-tâches obtenait des performances légèrement supérieures.

Nous en concluons donc qu'il est utile d'apprendre simultanément un mécanisme d'attention sur les EP et les discours haineux. De plus, cette étude vient renforcer les conclusions du chapitre précédent liées à l'importance des EP pour la détection des discours de haine.

8

Apprentissage par contraste pour la détection automatique des discours haineux

Dans ce chapitre, nous nous intéressons à l'apprentissage par contraste (*contrastive learning* en anglais) pour la détection des discours haineux. L'objectif de l'apprentissage par contraste est d'apprendre une représentation des données qui maximise la similarité entre des exemples positifs (appartenant à la même classe ou ayant des caractéristiques similaires) tout en minimisant la similarité entre des exemples négatifs (appartenant à des classes différentes ou ayant des caractéristiques différentes). Cet apprentissage a été très peu utilisé par les chercheurs du domaine de la détection des discours de haine. Dans ce chapitre, nous proposons une étude préliminaire sur l'efficacité de l'apprentissage par contraste pour notre tâche.

Pour la tâche de la détection des discours haineux, l'objectif de l'utilisation de l'apprentissage par contraste est d'apprendre des représentations de tweets qui : (1) rapprochent les plongements (*embeddings* en anglais) des tweets appartenant à la même classe et (2) éloignent les plongements des tweets appartenant à des classes différentes. Notre hypothèse est qu'un modèle de langage pré-entraîné avec cet apprentissage générera des plongements de tweets de meilleure qualité pour la détection des discours haineux. Ici, nous proposons d'étudier l'apprentissage par contraste avec un modèle de langage pré-entraîné. Nous utilisons une approche supervisée pour appliquer cette apprentissage et est fondée sur des paires de phrases. Pour cela, nous proposons deux approches pour la création des paires de tweets. Pour prédire les classes des tweets, nous proposons deux stratégies de décision fondées sur la similarité cosinus.

Nous articulons ce chapitre de la façon suivante : nous motivons nos travaux de recherche (section 8.1) et nous détaillons notre méthodologie pour l'utilisation de l'apprentissage par contraste (section 8.2) Puis, nous exposons les configurations expérimentales utilisées (section 8.3). Enfin, nous présentons les résultats obtenus (section 8.4) et nous concluons le chapitre (section 8.5).

Il est important de noter que ce chapitre contient une étude préliminaire de l'apprentissage par contraste pour la détection de la parole haineuse qui a débuté en fin de thèse.

8.1 Motivations

L'apprentissage par contraste est une méthode d'apprentissage automatique visant à apprendre les caractéristiques générales d'un ensemble de données en enseignant au modèle à identifier les similitudes ou les différences entre les points de données. Pour rappel, l'apprentissage

par contraste a émergé dans le domaine de la vision par ordinateur permettant d'améliorer la classification d'images (Hadsell et al., 2006; He et al., 2019). L'utilisation de cet apprentissage dans le domaine du traitement automatique des langues est récente (Rethmeier and Augenstein, 2023). L'apprentissage par contraste a notamment montré son efficacité pour la tâche de similarité sémantique textuelle (STS, pour *Semantic Textual Similarity* en anglais) (Reimers and Gurevych, 2019; Gao et al., 2021; Qi et al., 2022) ainsi que pour la tâche de classification de texte (Pappas and Henderson, 2019; Chen et al., 2022). Cependant, l'utilisation de cet apprentissage a été très peu exploré pour notre tâche (Kim et al., 2022; Shapiro et al., 2022; Zhong et al., 2022).

L'apprentissage par contraste peut être auto-supervisé ou supervisé. L'objectif de l'apprentissage par contraste auto-supervisé est d'apprendre des caractéristiques générales d'un ensemble de données afin de rendre un modèle robuste à l'injection de bruit. L'objectif de l'apprentissage par contraste supervisé est d'enseigner à un modèle à rapprocher des données similaires (appelées "positives") et à éloigner des données différentes (appelées "négatives"). La figure 8.1 illustre l'objectif de l'apprentissage par contraste supervisé. De plus, l'approche supervisé de cet apprentissage peu s'apparenter à de la classification de paires de phrases. Dans notre cas il est plus intéressant d'utiliser l'apprentissage supervisé afin de prendre en compte l'annotation en discours de haine. Donc pour notre tâche, l'objectif de cet apprentissage est d'apprendre des plongements de phrases (*sentence embeddings* en anglais) dont les tweets de la même classe soient proches et les tweets de classes différentes soient éloignés.

FIGURE 8.1 – Objectif de l'apprentissage par contraste. A gauche les représentations avant l'utilisation de cet apprentissage et à droite les représentations une fois l'application de cet apprentissage appliqué. Les carrés rouges et les cercles bleus représentent respectivement des exemples de deux classes. Les flèches vertes montrent le rapprochement des exemples de la même classe. A l'inverse, les flèches rouges signifient l'éloignement des exemples de classes différentes lors de l'apprentissage.

Notre hypothèse est que l'utilisation de cet apprentissage permettra à un modèle de langage de générer des représentations de tweets permettant de les regrouper en fonction de leurs classes. Cela permettrait d'améliorer la détection des discours de haine. Pour utiliser une telle méthode d'apprentissage, nous avons besoin de créer des paires positives (de la même classe) et négatives (de classes différentes). Des chercheurs ont montré qu'utiliser des paires positives et négatives trop difficiles ne permet pas au système de mieux généraliser (Cai et al., 2020). Pour la tâche de détection des discours de haine, une paire facile serait constituée de deux phrases de la même classe qui sont sémantiquement proches. Une paire difficile pour l'apprentissage par contraste serait formée de deux phrases de classes différentes qui sont sémantiquement très proches. De plus, l'apprentissage par contraste permet une meilleure compréhension des résultats (Zhang et al.,

2022). En effet, les plongements de phrases générés après l'application de cet apprentissage sont sensibles au calcul de distance vectoriel ce qui permet d'expliquer la classification des exemples. De ce fait, nous nous posons les questions de recherches suivantes :

— Est-il efficace d'appliquer l'apprentissage par contraste pour la détection des discours de haine ?

— Comment créer les paires de tweets pour cet apprentissage ?

8.2 Méthodologie proposée

Dans cette section, nous présentons la méthodologie proposée pour l'utilisation de l'apprentissage par contraste en vue d'améliorer la détection des discours haineux. Comme mentionné dans la section précédente, nous utilisons l'apprentissage par contraste de façon supervisé. Notre objectif est d'appliquer cet apprentissage à un modèle de langage afin que les plongements de phrases générés par celui-ci soit plus robustes pour la détection des discours de haine. Pour répondre aux questions de recherches posées précédemment, nous proposons et comparons deux méthodes de création de paires de tweets (voir section 8.2.1). Ensuite, nous décrivons le modèle neuronal proposé (voir section 8.2.2). Enfin, nous proposons deux stratégies de décision pour classifier les tweets (voir section 8.2.3).

8.2.1 Création des jeux de données pour l'apprentissage par contraste

Nous proposons deux méthodes pour la création des paires de tweets, car comme mentionné plus tôt certaines paires ne sont pas pertinentes pour l'apprentissage par contraste comme les paires trop faciles ou trop difficiles. Nous proposons donc deux approches pour la création des ensembles d'apprentissage et de validation. Pour la création de l'ensemble d'apprentissage, nous utilisons uniquement le jeu de données de l'apprentissage. En ce qui concerne la création des paires de tweets pour l'ensemble de validation, nous utilisons à la fois l'ensemble de validation et l'ensemble d'apprentissage. Nous avons fait ce choix car le modèle appris par contraste n'utilise que des tweets de l'ensemble d'apprentissage et c'est une tâche plus facile pour lui si au moins un tweet de la paire à prédire a été observée lors de l'apprentissage. Nous utilisons les tweets d'apprentissage pour prédire la classe d'un tweet de test (voir section 8.2.3), alors nous nous mettons dans la même configuration pour affiner les poids du modèle. De plus, lors du développement du modèle (présenté dans la section suivante) nous avons observé que celui-ci généralisait mieux avec cette méthode de création de paire pour l'ensemble de validation. Il est important de noter que les paires de tweets créées sont étiquetées de la manière suivante : "1 si les deux tweets appartiennent à la même classe et "0" sinon.

La première méthode consiste à associer à chaque tweet de l'ensemble d'apprentissage n tweets de la même classe (pour les paires positives) et n tweets de classes différentes (pour les paires négatives). Les tweets pour constituer les paires sont choisis aléatoirement. Nous nommons cette méthode $\mathbf{P_{aléa}}$ dans la suite du chapitre. Cette méthode est une façon standard de créer automatiquement des paires de tweets. Elle permet de représenter au minimum chaque tweet $n \times 2$ fois dans l'ensemble créé. Cependant, elle conserve la distribution déséquilibrée des classes. Cette méthode peut créer des paires de tweets très faciles ou difficiles à classifier pour un modèle neuronal.

Notre deuxième méthode consiste à créer un jeu de données de paires de tweets en sélectionnant les paires qui présentent un intérêt particulier en se basant sur la similarité cosinus. Nous choisissons des paires positives et négatives qui présentent un certain degré de difficulté en

fonction de leurs similarités cosinus. Nous rappelons que des paires jugées trop faciles ou trop difficiles ne sont pas des exemples d'apprentissage pertinents à donner au modèle neuronal (Cai et al., 2020).

Pour les paires positives, notre objectif est de sélectionner des paires de tweets dont les plongements des tweets sont éloignés (c'est-à-dire dont la similarité cosinus est faible). En revanche, pour les paires négatives, nous cherchons à sélectionner des paires dont les plongements sont proches (c'est-à-dire dont la similarité cosinus est élevée). Cette méthode de création des paires de tweets est basée sur la similarité cosinus entre les tweets. Nous nommons cette méthode $\mathbf{P_{sim}}$ dans la suite de ce chapitre. Tout d'abord, nous générons les plongements de tweets à l'aide de l'encodeur Universal Sentence Encoder (Cer et al., 2018). Nous calculons [26], avec ces plongements de phrases, la matrice de similarité cosinus correspondant à la création des paires d'apprentissage ou de validation. Cette matrice contient les scores de similarités cosinus pour toutes les paires possibles. Cette matrice nous permet de sélectionner des paires de tweets dans deux intervalles de scores de similarité cosinus que nous détaillons dans la section 8.3.1. Les deux intervalles de similarité cosinus permettent la sélection de paires avec une difficulté moyenne et difficile. Une paire de difficulté moyenne est formée de tweets dont le score de similarité cosinus est situé proche de 0,5 pour les paires positives et négatives. Pour rappel, une paire considérée comme difficile est formée de tweets dont le score de similarité cosinus est inférieur ou proche de 0 (car les tweets sont considérés comme opposés ou indépendants) pour les paires positives ou proches de 1 (car les tweet sont considérés comme colinéaires) pour les paires négatives. Pour chaque intervalle de scores de similarité cosinus, nous faisons attention à ce que les tweets apparaissent le plus équitablement possible dans l'ensemble de paires de tweets. C'est-à-dire, à chaque étape de sélection d'une paire de tweets dans un intervalle donné nous sélectionnons la paire dont les tweets occurrent le moins dans l'ensemble créé. Cela dépend également de la distribution des scores de similarités cosinus des différents jeux de données utilisées. L'objectif est de sélectionner des paires de tweets plus pertinentes pour l'apprentissage par contraste et de mieux généraliser. Le point fort de cette méthode est que nous obtenons une distribution équilibrée des classes pour l'apprentissage par contraste comparés à la méthode $\mathrm{P_{aléa}}$.

8.2.2 Systèmes pour l'apprentissage par contraste

Pour l'apprentissage par contraste, nous proposons d'utiliser l'approche basée sur le système SentenceBERT. Nous rappelons que le système SentenceBERT est basé sur un modèle de langage BERT siamois et a été proposé pour la tâche de similarité entre deux phrases. Dans notre cas, nous adoptons l'approche présentée par Reimers and Gurevych (2019) pour appliquer l'apprentissage par contraste sur des paires de tweets. Nous désignons ce système sous le nom de **sBERT-ACH** (pour *SentenceBERT utilisant l'Apprentissage par Contraste pour la Haine*). Le système sBERT-ACH utilise les jeux de paires de tweets que nous avons préalablement créés (section 8.2.1). La figure 8.2 illustre l'apprentissage du système sBERT-ACH à partir d'un exemple de paire de tweets. Le système est fondé sur le modèle de langage BERT, et il est essentiel de noter que les poids du modèle de langage BERT sont mis à jour lors de l'apprentissage par contraste. Le modèle de langage BERT est partagé entre les deux entrées, c'est-à-dire les tweets a et b, dans la figure 8.2. Le système utilise les plongements de phrases (les plongements du jeton [CLS]) pour apprendre si les tweets a et b appartiennent à la même classe ou à des classes différentes (représenté par la *Contrastive Loss* dans la figure 8.2). L'objectif de la *Contrastive Loss* est de rapprocher les plongements des paires de tweets positives (étiquetées 1) et d'éloigner les

26. En utilisant la fonction *cosine_similarity* de la librairie python *Sklearn*

plongements des paires de tweets négatives (étiquetées 0) (Hadsell et al., 2006).

La fonction de perte *Contrastive Loss* pour un ensemble P de paires de tweets est définie comme ci-dessous (tirée de l'article de Hadsell et al. (2006)) :

$$\mathcal{L} = \sum_{i=1}^{\mathcal{P}} (Y)\mathcal{L}_S + (1-Y)\mathcal{L}_D \tag{8.1}$$

où

$$\mathcal{L}_S = \frac{1}{2}(sim(\mathbf{z}_u, \mathbf{z}_v))^2 \tag{8.2}$$

et

$$\mathcal{L}_D = \frac{1}{2}(max\{0, m - sim(\mathbf{z}_u, \mathbf{z}_v)\})^2 \tag{8.3}$$

Y représente l'étiquette de la paire (u, v) dont les plongements de phrases sont $\mathbf{z}_u$ et $\mathbf{z}_v$ et $\mathcal{P}$ correspond au nombre de paires. La fonction $sim(u, v)$ correspond à la distance cosinus $((\mathbf{z}_u^T \mathbf{z}_v)/(|\mathbf{z}_u||\mathbf{z}_v|))$ entre les plongements $\mathbf{z}_u$ et $\mathbf{z}_v$. Les fonctions de pertes $\mathcal{L}_S$ et $\mathcal{L}_D$ représentent respectivement les fonctions de pertes pour les paires positives et négatives. Le paramètre m représente la valeur de la marge pour éloigner les plongements des paires négatives. Cette fonction de perte vise à minimiser la similarité cosinus entre les tweets des paires négatives et à maximiser la similarité cosinus des tweets des paires positives.

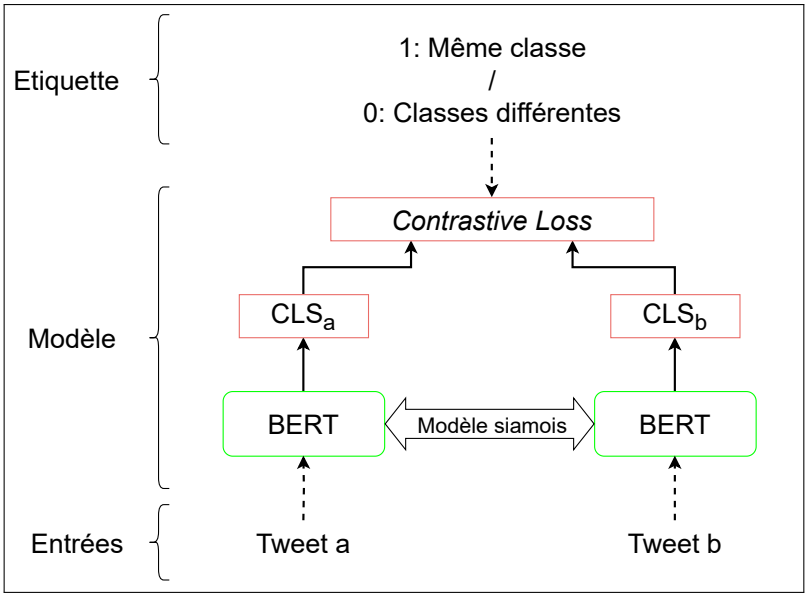


FIGURE 8.2 – Apprentissage du système sBERT-ACH sur les paires de tweets. *Tweet a* et *Tweet b* représente une paire de tweets utilisée en entrée. La flèche d'équivalence logique signifie que le modèle BERT est identique pour les deux entrées.

La figure 8.3 illustre l'utilisation du système sBERT-ACH pour la classification des tweets de test. Pour cela, nous générons les plongements de phrase (les plongements du jeton [CLS]) à partir du système sBERT-ACH pour chaque tweet de test et d'apprentissage. Ensuite, nous calculons les scores de similarité cosinus entre les plongements de test et ceux de l'ensemble d'apprentissage. Nous utilisons l'ensemble d'apprentissage pour calculer les scores de similarité cosinus, car nous avons besoin d'un ensemble de données annotées manuellement pour classifier un tweet de test.

Cette stratégie de décision des tweets de test est fortement inspirée du système de base, c'est-à-dire SentenceBERT. Il est important de noter que SentenceBERT a été initialement utilisé pour la tâche de relation entre deux phrases, qui vise à déterminer si deux phrases sont liées ou non. Dans notre cas, nous appliquons une méthode de décision (introduite dans la section 8.2.3) pour classifier les tweets de test en se basant sur les scores de similarité cosinus obtenus entre l'ensemble de test et l'ensemble d'apprentissage.

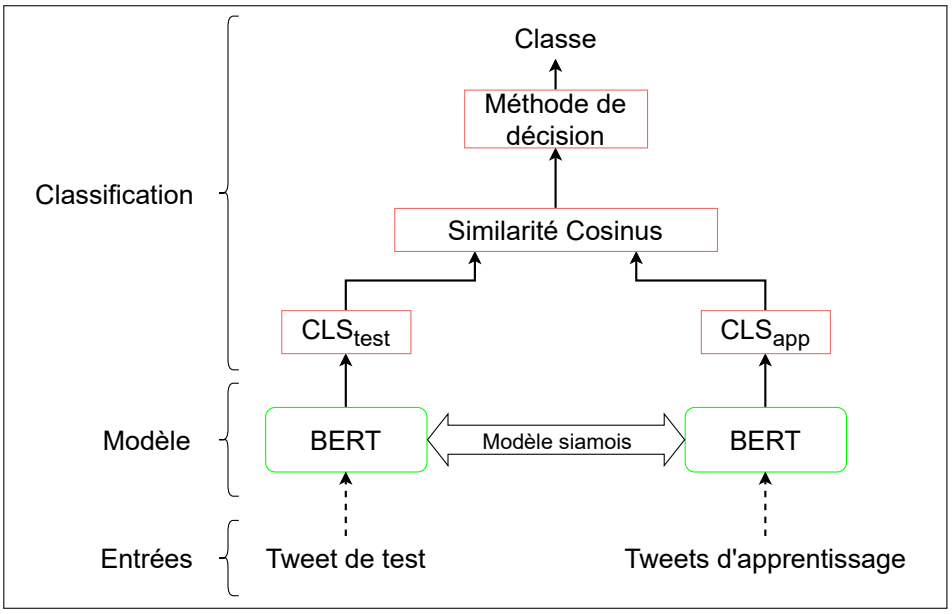


FIGURE 8.3 – Classification des tweets de test avec le système sBERT-ACH en utilisant les tweets d'apprentissage.

Nous comparons notre approche avec deux autres systèmes : BERT-FT et DualCL. Le système BERT-FT (pour *BERT fine-tuned*) est un système de classification classique basé sur le modèle de langage BERT. Il utilise les plongements des tweets (jeton [CLS]) pour apprendre à classifier les tweets. Les poids du modèle de langage BERT sont affinés pendant l'apprentissage de ce système. Il est important de noter que ce système n'utilise pas l'apprentissage par contraste.

Le système DualCL (pour *Dual Contrastive learning*), proposé par Chen et al. (2022), est fondé sur le modèle de langage BERT. Il utilise à la fois l'apprentissage supervisé pour la classification des tweets avec la fonction de perte d'entropie croisée, et l'apprentissage par contraste en utilisant une augmentation de données avec des étiquettes. Pour cela, il ajoute des jetons pour représenter les différentes classes dans chaque phrase : entre le jeton [CLS] et le premier mot de la phrase. L'objectif de leur approche est de rapprocher les jetons [CLS] des phrases de l'étiquette correspondante à celle-ci. Chen et al. (2022) proposent une double fonction de perte pour l'apprentissage par contraste, appelée *dual loss*. Cette fonction de perte possède deux objectif : le premier vise à rapprocher les représentations des tweets de la même classe, tandis que le second consiste à éloigner les représentations des tweets de classes différentes. La perte finale est calculée en combinant l'entropie croisée et la *dual loss*.

En comparant expérimentalement notre approche avec ces deux systèmes, nous évaluons les performances de l'apprentissage par contraste pour la détection des discours haineux par rapport à un système classique (BERT-FT) et une autre approche (DualCL) qui utilise à la fois l'apprentissage par contraste et l'apprentissage de classification.

8.2.3 Méthodes de décision pour la classification des tweets

Pour classifier un tweet de test nous utilisons les tweets de l'ensemble d'apprentissage. Nous admettons que nos propositions sont difficilement applicables dans le monde de l'industrie, cependant elles permettent une d'expliquer la prédiction des classes. En effet, nous proposons deux méthodes fondées sur la distance cosinus dont les prédictions sont facilement compréhensibles comparés aux prédictions de classifieurs fondés sur des réseaux neuronaux. Pour prédire les classes de haine des tweets de test avec le système sBERT-ACH, nous proposons deux méthodes basées sur la similarité cosinus entre les plongements de phrases (générés par le modèle de langage entraîné avec l'apprentissage par contraste) des ensembles de test et d'apprentissage.

La première méthode, nommée *Moy*, consiste à prendre le score de similarité cosinus le plus élevé obtenu en calculant la moyenne des scores de similarité cosinus pour chacune des classes d'un jeu de données entre un tweet de test et les tweets de l'ensemble d'apprentissage de cette classe. La classe qui produit la plus grande moyenne de similarités cosinus est choisie comme classe prédite pour le tweet de test. Cet méthode de décision est définie par l'équation suivante :

$$C_{test} = argmax_{c=1}^{C}(\frac{1}{|N_c|}\sum_{k=1}^{N_c} sim(T_{test}, T_{app}^{k}))$$

où C et N_c représentent respectivement le nombre de classes et le nombre de tweets d'apprentissage de la classe c. T_{test} et C_{test} représentent respectivement le tweet de test et sa classe prédite. T_{app} représentent les tweets de l'ensemble d'apprentissage. Pour rappel, la fonction sim correspond à la similarité cosinus.

La deuxième méthode consiste à effectuer un vote majoritaire parmi un ensemble de tweets d'apprentissage les plus proches, selon la distance cosinus, d'un tweet de test. Pour cette méthode de décision, nous récupérons les K tweets d'apprentissage les plus similaires d'un tweet de test (voir équation 8.4) et d'utiliser la classe majoritaire dans cette ensemble K comme classe prédite pour le tweet de test (voir équation 8.5). Nous nommons cette méthode de décision *VM* (pour vote majoritaire) dans la suite du chapitre.

$$K \text{ plus proches tweets} = \max_{k=1}^{K}\text{sim}(T_{\text{test}}, T_{\text{app}}^{k}) \tag{8.4}$$

$$C_{test} = \text{argmax}_c \sum_{k=1}^{K}[C_{\text{app}}^{k} = c] \tag{8.5}$$

8.3 Configurations expérimentales

8.3.1 Création des jeux de paires de tweets

Pour la création des jeux de paires de tweets, nous utilisons deux méthodes $\text{P}_{\text{aléa}}$ et P_{sim} introduites dans la section 8.2.

Concernant la méthode $\text{P}_{\text{aléa}}$ nous créons 50 paires de tweets par classe pour chaque tweet de l'ensemble d'apprentissage. La valeur de 50 a été optimisé sur l'ensemble de validation de Waseem. Pour créer une paire de l'ensemble de validation, nous prenons un tweet de l'ensemble de validation et un tweet de l'ensemble d'apprentissage. Pour chaque tweet de l'ensemble de validation, nous sélectionnons aléatoirement 10 tweets de chacune des classes de l'ensemble d'apprentissage.

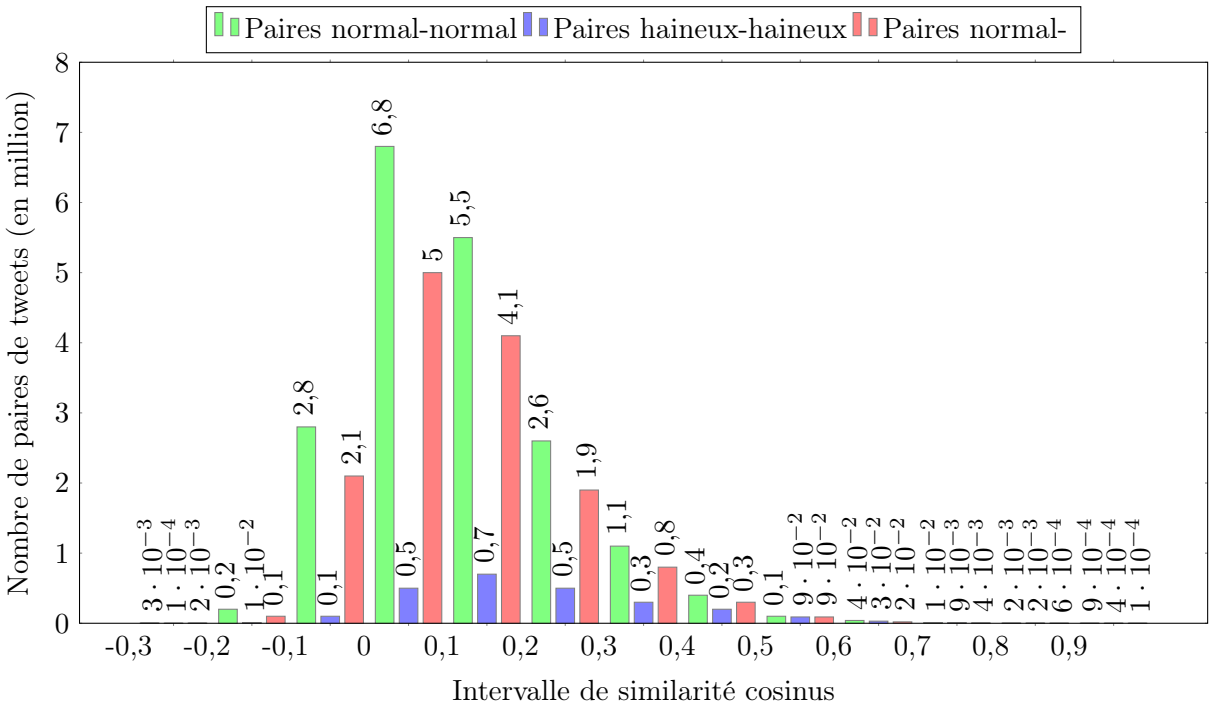


FIGURE 8.4 – Distribution des paires de tweets de l'ensemble d'apprentissage de Waseem selon la similarité cosinus. Les paires positives normales et haineuses sont respectivement formées de deux tweets normaux ou haineux. Les paires négatives sont formées d'un tweet normal et d'un autre haineux.

Pour la méthode de P_{sim}, nous sélectionnons les paires de tweets en fonction de plusieurs intervalles de similarité cosinus. Le choix des différents intervalles de similarité cosinus est basé sur les distributions obtenues pour les paires de tweets selon les intervalles de similarité cosinus. La figure 8.4 présente la distribution de toutes les paires de tweets possibles d'apprentissage du jeu de données de Waseem. Comme mentionné dans notre méthodologie, nous sélectionnons des paires de tweets dans deux intervalles pour les paires positives et négatives. Pour rappel, nous faisons attention à équilibrer la distribution des classes de paires de tweets. Cependant, le choix des intervalles varie en fonction des jeux de données et du nombre de paires sélectionnées dans ceux-ci, car il faut un nombre suffisant de paires de tweets dans ces intervalles.

Paires de tweets		**Classification binaire Waseem - HatEval** 50k - 100k - 150k	**Classification ternaire Davidson** 50k	100k	150k	**Founta** 50k - 100k - 150k
Positives	moyennes	[0,5 ; 0,7]	[0,4 ; 0,8]	[0,3 ; 0,8]		[0,5 ; 0,7]
	difficiles	[0,1 ; 0,3]	[0,1 ; 0,3]	[0,05 ; 0,3]	[0,01 ; 0,3]	[0,1 ; 0,3]
Négatives	moyennes	[0,3 ; 0,4]	[0,3 ; 0,4]	[0,2 ; 0,4]		[0,4 ; 0,6]
	difficiles	[0,4 ; 0,9]	[0,4 ; 0,9]			[0,6 ; 0,9]

TABLE 8.1 – Choix des intervalles de similarités cosinus pour la création des ensembles d'apprentissage avec la méthode P_{sim}50k, P_{sim}100k et P_{sim}150k.

Nous avons décidé d'évaluer cette méthode de sélection des paires de tweets en utilisant

différentes valeurs de n, à savoir *n=(50000, 100000, 150000)*, que nous avons respectivement nommées P_{sim}50k, P_{sim}100k et P_{sim}150k dans la suite du chapitre. Dans la table 8.1, nous présentons les intervalles de similarités cosinus utilisées sur chacun des jeux de données pour la création des paires de tweets avec la méthode P_{sim}. Nous avons choisi ces intervalles de scores de similarité cosinus car ils contiennent suffisamment d'exemples de paires de tweets pour constituer un jeu de données homogène en termes de nombre de paires positives et négatives, tout en maintenant un niveau de difficulté pour l'apprentissage par contraste. Concernant le jeu de données Davidson nous avons dû compléter l'ensemble d'apprentissage en augmentant l'intervalle de sélection des paires positives (de difficultés moyennes et difficiles) et négatives moyennes. En effet, pour Davidson nous avons remarqué un manque de paires de tweets dans ces intervalles et cela est notamment dû au fait du déséquilibre de la distribution des classes. Nous évitons ainsi de prendre des paires positives dont les tweets sont trop similaires et inversement pour les paires négatives, car ce ne sont pas des paires intéressantes comme mentionné précédemment (voir section 8.2). Le nombre de paires de tweets des différents ensembles d'apprentissage est exposée dans le tableau 8.2.

Méthodes de création	**Nombre de paires de tweets (en million)**			
	Waseem	**HatEval**	**Davidson**	**Founta**
$P_{aléa}$ 50	0,86	0,90	2,6	7,6
P_{sim}50k	0,40		0,60	
P_{sim}100k	0,80		1,20	
P_{sim}150k	1,20		1,80	

TABLE 8.2 – Nombre de paires de tweets sélectionnées (en million) avec les méthodes $P_{aléa}$ et P_{sim} pour créer les ensembles d'apprentissage.

Dans la figure 8.5, nous illustrons la différence de distribution, en termes de paires positives et négatives, entre la méthode $P_{aléa}$ et P_{sim} (P_{sim}100k). Avec la méthode P_{sim}, nous obtenons un jeu de paires de tweets équilibré entre les différentes classes de paires de tweets (normal-normal, haineux-haineux et normal-haineux), contrairement à la méthode $P_{aléa}$ qui conserve le déséquilibre des classes contenu dans le jeu de données. Ainsi, la distribution des tweets de classes *normal* et *haineux* est maintenue dans le jeu de données de paires de tweets (70% de tweets de la classe normale et 70% de paires de tweets positives normales).

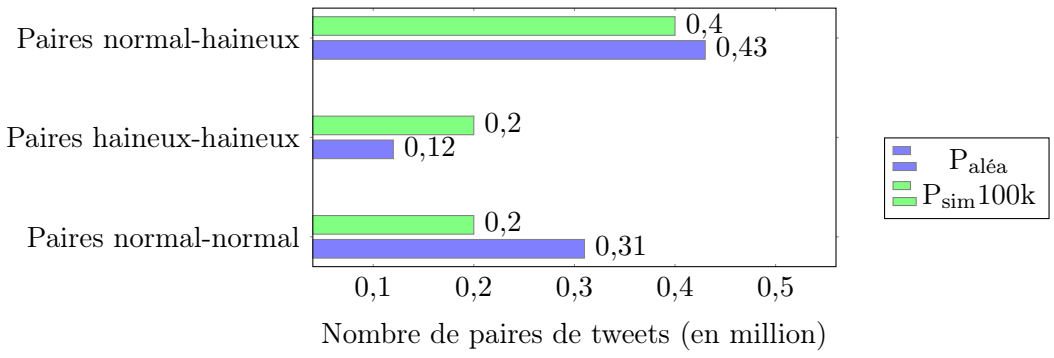


FIGURE 8.5 – Distribution des paires de tweets sur l'ensemble d'apprentissage de Waseem pour les méthodes de création de $P_{aléa}$ et P_{sim}100k.

8.3.2 Paramètres des systèmes

Pour l'apprentissage du système sBERT-ACH, nous utilisons un taux d'apprentissage (*learning rate* en anglais) de 2×10^{-5}, qui est ajusté après chaque étape d'évaluation. Pour cela, nous utilisons un optimiseur de type *Adam*. Nous entraînons le système sBERT-ACH sur une époque d'apprentissage, car l'apprentissage sur plusieurs époques n'a pas montré d'amélioration significative lors du développement du système. Nous effectuons 5 étapes d'évaluation pendant l'époque d'entraînement. Le système sBERT-ACH est entraîné sur les paires de tweets créées selon les méthodes décrites dans la section précédente (section 8.3.1).

Concernant le système DualCL, nous utilisons les paramètres proposés par (Chen et al., 2022). Le taux d'apprentissage est de 10^{-5}. Nous effectuons l'entraînement sur 100 époques, avec un arrêt précoce (*early stopping* en anglais) de 5 époques de patience. L'arrêt précoce est basé sur la diminution du taux de perte sur l'ensemble de validation. Nous sélectionnons le modèle qui atteint le taux de perte le plus bas sur l'ensemble de validation.

Pour le système BERT-FT, nous utilisons un taux d'apprentissage initialisé à 10^{-5}. Comme pour le système DualCL, le système BERT-FT est entraîné sur 100 époques, avec un arrêt précoce après 5 époques de patience basé sur le taux de perte sur l'ensemble de validation. Les systèmes DualCL et BERT-FT sont entraînés sur les jeux de données pour la classification des discours haineux introduits dans le chapitre 3 et ces modèles utilisent un optimiseur de type *Adam* comme notre système.

Le nombre d'exemples traités simultanément (*batch size* en anglais) pour tous les systèmes est de 16. Comme précédemment, nous évaluons les systèmes avec la mesure d'évaluation *macro-F1* introduite dans le chapitre 3.1.7.

8.4 Résultats obtenus

Tout d'abord, nous comparons nos deux méthodes de décision (pour classifier les tweets de test) utilisées avec sBERT-ACH entraîné sur le jeu de paires de tweets $P_{\text{aléa}}$. Ensuite, nous comparons les deux méthodes de création de jeux de paires de tweets. Enfin, nous comparons les performances des systèmes utilisant l'apprentissage par contraste avec celles du système de base entraîné sur la classification des tweets.

8.4.1 Comparaison des méthodes de décision

La figure 8.6 présente les résultats obtenus avec les deux méthodes de décision des classes, Moy et VM, en utilisant le système sBERT-ACH entraîné sur le jeu de paires de tweets construit avec la méthode $P_{\text{aléa}}$. Les résultats sont présentés en terme de *macro-F1* en moyenne sur les quatre ensembles de test. Pour la méthode VM, nous avons testé plusieurs valeurs pour le paramètre K, qui correspond au nombre de tweets ayant le score de similarité cosinus le plus élevé. Les valeurs testées sont 1, 3, 5, 7, 9 et 15. Nous observons que l'utilisation de la méthode Moy (73,9% de score de *macro-F1*) surpasse significativement la méthode VM (scores de *macro-F1* compris entre 70,9% et 72,7%). Cela est probablement dû au déséquilibre de la distribution des classes dans les jeux de données, car il y a plus de probabilité qu'il existe des tweets d'apprentissage de la classe majoritaire qui soient proches d'un tweet de test. Par exemple, nous rappelons que Waseem possède 73% de tweets normaux contre 27% de tweets haineux. La méthode Moy permet de lisser les scores de similarités sur tous les tweets d'apprentissage d'une classe contrairement à la méthode VM. Il est important de noter que pour la méthode VM, le score de *macro-F1*

maximum est obtenu avec $K = 15$, et qu'un plateau est atteint pour $K > 15$. Dans la suite nous utiliserons uniquement la méthode de décision Moy.

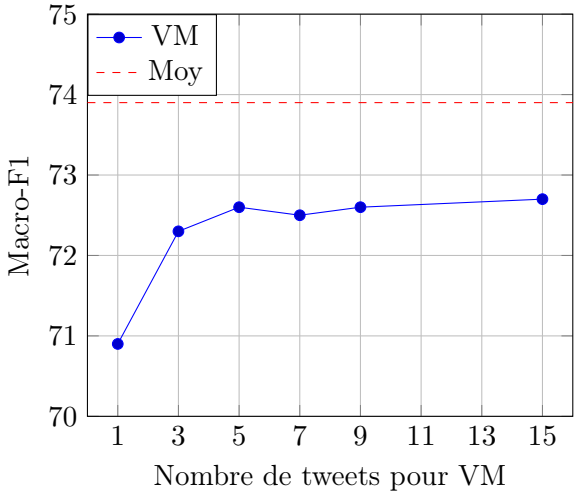


FIGURE 8.6 – Comparaison des scores *macro-F1* médians (moyenne sur quatre jeux de données) obtenus avec les méthodes de décision VM et Moy pour le système sBERT-ACH. Nombre de tweets pour VM (abscisse) représente le nombre de tweets sélectionnés avec le plus haut score de similarité cosinus pour la méthode VM.

8.4.2 Comparaison des méthodes de créations de paires de tweets

La figure 8.7 présente les scores de *macro-F1* obtenus sur les ensembles de test avec le système sBERT-ACH en utilisant différentes méthodes de création des jeux de paires de tweets :$P_{aléa}$, P_{sim}50k, P_{sim}100k et P_{sim}150k. Nous rappelons que ces résultats sont obtenus en utilisant la méthode de décision Moy.

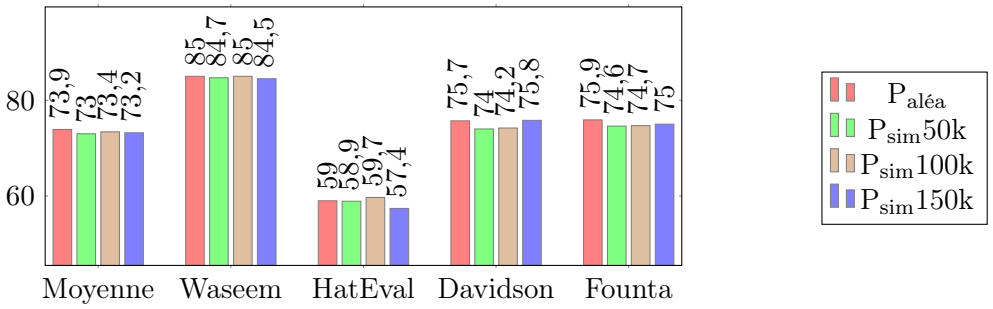


FIGURE 8.7 – Scores *macro-F1* médians du système sBERT-ACH avec les différentes méthodes de création des ensembles d'apprentissage.

Nous constatons que le système sBERT-ACH utilisant la méthode $P_{aléa}$ obtient le meilleur score *macro-F1* en moyenne sur tous les ensembles de test. Cependant l'amélioration n'est pas significative comparé à la méthode P_{sim}, selon le test de paires appariées avec un risque de 5% (Gillick and Cox, 1989). Lorsqu'il est entraîné sur le jeu de paires de tweets $P_{aléa}$, sBERT-ACH atteint une moyenne de 73,9% de score *macro-F1*, comparé à 73%, 73,4% et 73,2% respectivement

pour les entraînements sur P_{sim}50k, 100k et 150k. Nous notons que P_{sim} génère moins de paires de tweets que lla méthode $P_{aléa}$ (voir la table 8.2). Nous constatons sur Waseem et HatEval qu'utiliser les jeux de paires de tweets P_{sim}150k obtient des scores de *macro-F1* légèrement inférieurs (significatif sur HatEval) à l'utilisation des jeux générés avec la méthode $P_{aléa}$. Concernant les jeux de données Davidson et Founta, nous remarquons que plus nous augmentons le nombre de paires de tweets avec la méthode P_{sim}, plus le score de *macro-F1* augmente. Sur Davidson nous observons 74%, 74,2% et 75,8% de scores de *macro-F1* médian pour respectivement P_{sim}50k, 100k et 150k.

Nous concluons que l'utilisation de moins de paires de tweets avec la méthode P_{sim} permet au système sBERT-ACH d'obtenir des résultats similaires à ceux obtenus avec les paires de tweets générées avec la méthode $P_{aléa}$.

8.4.3 Comparaison des apprentissages

La table 8.3 présente les scores médians de *macro-F1* obtenus sur les ensembles de test des jeux de données Waseem, HatEval, Davidson et Founta.

Systèmes	**Classification binaire**		**Classification ternaire**		**Moyenne**
DPH	**Waseem**	**HatEval**	**Davidson**	**Founta**	
BERT-FT	84,3 ($\pm 0,5$)	59,7 ($\pm 2,6$)	75,5 ($\pm 4,4$)	75,4 ($\pm 0,7$)	73,7
DualCL	84,3 ($\pm 0,9$)	**63,0** ($\pm 2,0$)	73,5 ($\pm 3,6$)	74,6 ($\pm 1,2$)	73,8
sBERT-ACH$_{P_{aléa}}$	**85,0** ($\pm 0,5$)	59,0 ($\pm 2,1$)	75,7 ($\pm 0,5$)	**75,9** ($\pm 0,5$)	**73,9**
sBERT-ACH$_{P_{sim}150k}$	84,5 ($\pm 0,6$)	57,4 ($\pm 1,8$)	**75,8** ($\pm 0,1$)	75,0 ($\pm 0,2$)	73,2

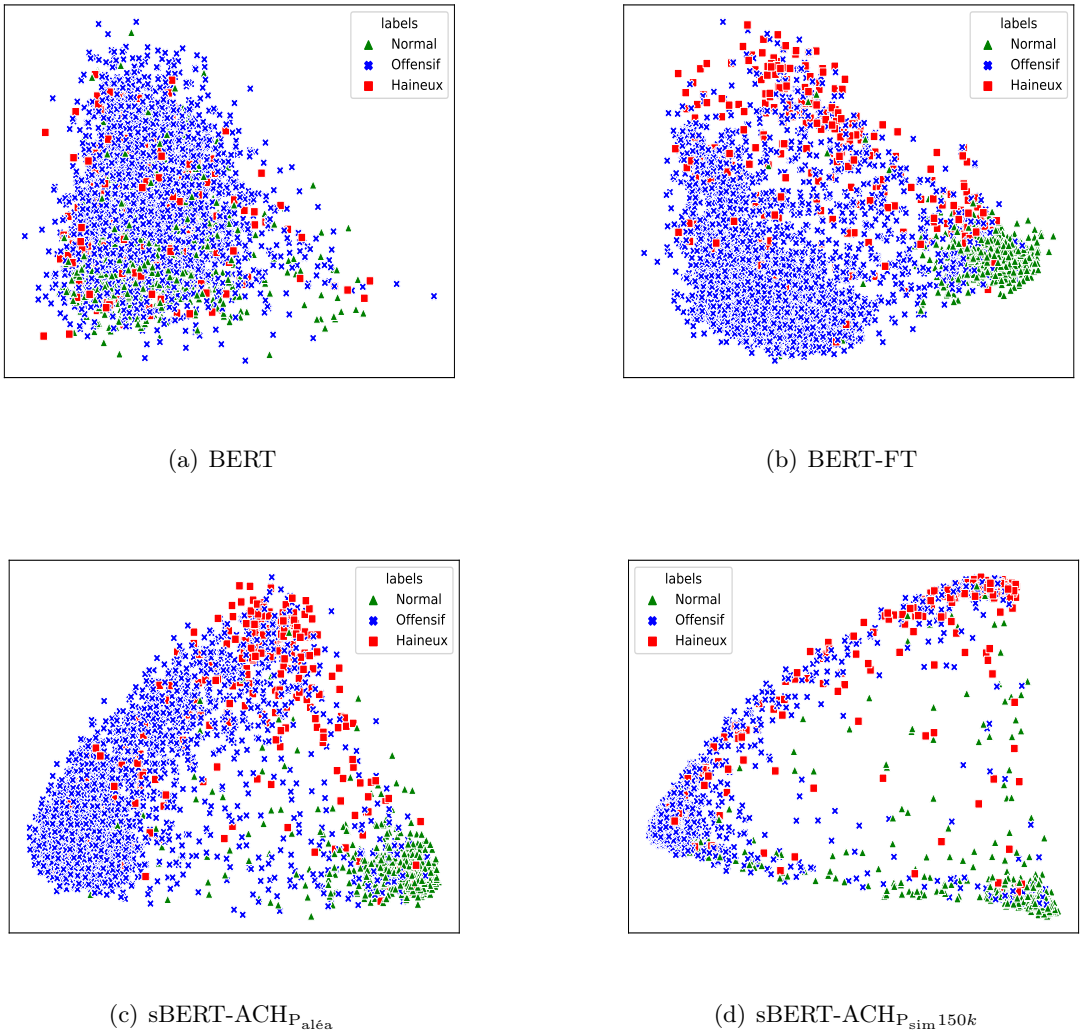
TABLE 8.3 – *Macro-F1* médians et écarts types sur les ensembles de test. Les résultats soulignés représente une amélioration significative comparé au modèle BERT-FT qui n'utilise pas l'apprentissage par contraste. La colonne *Moyenne* représente la moyenne des scores médians sur les quatre jeux de données et l'amélioration significative est calculée en concaténant toutes les prédictions. Les meilleurs résultats sont en gras pour chacun des ensembles de test.

Nous constatons que le système *BERT-FT* obtient un score de *macro-F1* moyen de 73,7%. Nous remarquons que le système DualCL affiche des résultats similaires à BERT-FT, avec une moyenne de 73,8% de *macro-F1*. Nous notons que le système DualCL surpasse significativement BERT-FT uniquement sur HatEval (63% contre 59,7%). Nous notons que le système DualCL utilise deux fonctions de perte lors de l'apprentissage, à savoir l'entropie croisée et la perte contrastive. Concernant notre approche par paires de tweets, nous constatons qu'avec le jeu de paires de tweets $P_{aléa}$, le système sBERT-ACH obtient un score de *macro-F1* moyen de 73,9%. En comparaison avec le système *BERT-FT*, ce système améliore les scores de *macro-F1* sur les jeux de données Waseem, Davidson et Founta. Notre approche obtient respectivement 85,0%, 75,7% et 75,9% de scores *macro-F1* sur Waseem, Davidson et Founta, tandis que le système *BERT-FT* obtient 84,3%, 75,5% et 75,4%.

En ce qui concerne notre approche avec P_{sim}150k, nous remarquons que le système sBERT-ACH$_{P_{sim}150k}$ obtient des résultats légèrement inférieurs (non significatifs) par rapport aux systèmes BERT-FT, DualCL et sBERT-ACH$_{P_{aléa}}$, avec une moyenne de 73,2% de score *macro-F1* sur tous les ensembles de test. Cela est probablement dû au fait que sBERT-ACH$_{P_{sim}150k}$ utilise beaucoup moins de paires de tweets. Nous constatons que sélectionner moins de paires de tweets pour l'apprentissage par contraste peut être effectué avec notre méthode P_{sim} sans dégrader significativement la détection des discours de haine.

8.4.4 Visualisation des plongements des tweets

Ici, nous analysons les plongements des tweets en les projetant dans un espace à deux dimensions à l'aide de l'analyse en composantes principales, afin de réduire la dimension des plongements dans un espace visualisable. La figure 8.8 présente les projections des plongements des tweets de l'ensemble de validation de Davidson, réalisées avec les systèmes BERT (non entraîné), BERT-FT, sBERT-ACH$_{\mathrm{P_{aléa}}}$ et sBERT-ACH$_{\mathrm{P_{sim}}150k}$.



(a) BERT

(b) BERT-FT

(c) sBERT-ACH$_{\mathrm{P_{aléa}}}$

(d) sBERT-ACH$_{\mathrm{P_{sim}}150k}$

FIGURE 8.8 – Projections par analyse en composantes principales des plongements de tweets, de l'ensemble de validation de Davidson, générés par les systèmes BERT (a), BERT-FT (b), sBERT-ACH$_{\mathrm{P_{aléa}}}$ (c) et sBERT-ACH$_{\mathrm{P_{sim}}150k}$ (d).

Nous remarquons qu'avant l'affinage du modèle BERT (voir figure 8.8(a)), les régions correspondantes à deux classes différentes se superposent et il est difficile de distinguer des regroupements clairs correspondant à différentes classes. Avec le système BERT-FT (voir figure 8.8(b)), on observe deux regroupements distincts : un groupe de triangles verts pour les tweets normaux et un autre regroupant les tweets offensifs et haineux. En utilisant l'apprentissage par contraste

sBERT-ACH$_{\text{P}_{\text{aléa}}}$ (voir figure 8.8(c)), il y a une meilleure séparation entre les tweets normaux (groupe vert) et les tweets offensifs (groupe bleu), avec en plus l'apparition d'un troisième regroupement en haut de la figure pour les tweets haineux. De plus, le système sBERT-ACH$_{\text{P}_{\text{sim}}150k}$ (voir figure 8.8(d)) montre des regroupements plus distincts entre les tweets normaux (groupe vert), les tweets offensifs (groupe bleu) et les tweets haineux (groupe rouge) par rapport au système sBERT-ACH$_{\text{P}_{\text{aléa}}}$. Il est important de rappeler que le système sBERT-ACH$_{\text{P}_{\text{sim}}150k}$ utilise environ 30% de paires de moins que le système sBERT-ACH$_{\text{P}_{\text{aléa}}}$. Pour conclure, nous observons que l'apprentissage par contraste permet d'obtenir une meilleure séparation des classes.

8.5 Conclusion du chapitre

Dans ce chapitre, nous avons étudié l'utilisation de l'apprentissage par contraste pour la détection des discours haineux. Nous avons proposé le système sBERT-ACH, basé sur l'architecture du système SentenceBERT, qui utilise l'apprentissage par contraste sur des paires de tweets.

Nous avons présenté deux méthodes pour la construction de paires de tweets pour l'apprentissage par contraste, $\text{P}_{\text{aléa}}$ et P_{sim}, ainsi que deux méthodes de décision : avec la moyenne des scores de similarité cosinus et par vote majoritaire. Nous avons observé que la méthode de décision avec la moyenne des similarités cosinus surpassait la méthode de décision par vote majoritaire lorsque nous utilisions l'apprentissage par contraste sur les paires de tweets créées avec la méthode $\text{P}_{\text{aléa}}$. De plus, nous avons constaté que les paires de tweets créées avec la méthode P_{sim} permettaient d'obtenir des résultats similaires à ceux obtenus avec la méthode $\text{P}_{\text{aléa}}$ en utilisant un nombre moins élevé de paires de tweets pour l'apprentissage par contraste.

Nous avons également comparé notre approche basée sur l'apprentissage par contraste avec l'entraînement classique du modèle de langage BERT. Nous avons observé des résultats similaires entre les deux approches. Cependant, nous avons observé qu'avec l'apprentissage par contraste, nous obtenons une meilleure séparation entre les classes de tweets, qui ne s'est pas traduite par une amélioration significative de la détection des discours de haine avec nos méthodes de décision.

9

Conclusions et perspectives

L'augmentation exponentielle de l'utilisation des réseaux sociaux a engendré une prolifération des messages néfastes, tels que les discours de haine. Cette thèse apporte des contributions pour la détection de contenus haineux dans le réseau social de Twitter en proposant différentes approches fondées sur les réseaux neuronaux. Nous avons étudié l'intégration de différentes caractéristiques dans un réseau neuronal fondé sur les plongements de phrases (*sentence embeddings* en anglais). De plus, nous avons analysé l'impact des expressions polylexicales (EP) pour la détection des discours de haine. Afin de mener à bien notre recherche, nous avons étudié la robustesse de systèmes d'identification d'EP dans les tweets. Nous avons également étudié l'apport de deux méthodes d'apprentissage, tels que l'apprentissage multi-tâches et par contraste dans le cadre de la détection des discours de haine.

Dans ce chapitre, nous résumons les principales contributions de cette thèse (voir la section 9.1). Nous proposons plusieurs perspectives de recherche, à court et long termes (voir la section 9.2).

9.1 Résumés des contributions

Dans le **chapitre 4**, nous avons étudié l'impact de l'intégration de différentes caractéristiques linguistiques dans un réseau neuronal utilisant les plongements de phrases Les caractéristiques linguistiques sont les suivantes : la ponctuation, la casse des mots, les termes haineux, les émojis et les parties du discours (*part-of-speech* en anglais). De plus, nous avons proposé un modèle neuronal utilisant deux branches : la première branche neuronale est dédié aux plongements de phrases et la deuxième branche intègre les caractéristiques linguistiques. Voici nos conclusions :

— Nous avons observé que l'ajout des caractéristiques (ponctuation, casse des mots, termes haineux et parties du discours) n'amélioraient pas les résultats du système fondé sur les plongements de phrases.

— Nous avons constaté que les émojis apportent une information supplémentaire au système de base utilisant uniquement les plongements de phrases. De plus, nous avons remarqué une amélioration, soit en intégrant les émojis sous forme textuelle, soit en utilisant des plongements calculés à partir des émojis. En effet, nous avons observé que certains émojis étaient plus utilisés dans les discours haineux que dans des tweets normaux et inversement.

Après cette étude, nous nous sommes consacré à l'intégration des EP dans un système de détection des discours haineux. Cependant, pour utiliser les EP dans les jeux de données pour la

détection des discours de haine, nous avons dû dans un premier temps les identifier. Pour rappel, la tâche d'identification des EP est complexe et a été très peu étudiée dans les tweets.

Dans le **chapitre 5**, nous avons étudié la robustesse de deux systèmes d'identification d'EP. Le premier système utilise un dictionnaire d'EP. Ce dictionnaire est créé automatiquement à partir de jeux de données annotés manuellement. Le deuxième système est fondé sur des réseaux neuronaux profonds. Concernant le système neuronal, nous avons étudié différentes configurations en variant les ensembles d'apprentissage et les schémas d'étiquetage des EP. De plus, nous avons proposé une approche en deux étapes combinant le système neuronal et le système fondé sur le dictionnaire. Voici les différentes conclusions :

— Nous avons observé que le système neuronal surpasse le système à base de dictionnaire.

— Nous avons constaté qu'entraîner le système neuronal avec des données de tweets et des données provenant d'autres sources obtenait de meilleurs résultats sur la tâche d'identification des EP dans les tweets.

— Nous avons remarqué que l'utilisation des étiquettes de super-sens des mots et des catégories des EP dans les schémas d'étiquetage des EP n'apporte aucune amélioration des performances.

— Notre approche en deux étapes surpasse le système neuronal et celui utilisant un dictionnaire d'EP.

Après cette étude, nous nous sommes focalisés sur l'impact de l'intégration des EP sur la détection des discours de haine. Nous nous sommes posés plusieurs questions de recherche concernant cette problématique :

(1) L'intégration des EP dans un système neuronal est-elle utile pour la détection de la parole haineuse ?

(2) Quel système d'identification d'EP est le plus performant pour la détection de la parole haineuse ?

(3) Est-il utile de faire attention aux EP et aux discours de haine simultanément ?

Dans le **chapitre 6**, nous avons répondu aux deux premières questions de recherche. Nous avons proposé deux systèmes intégrant les EP. Le premier système est fondé sur trois branches neuronales. Dans la première branche, le système utilise les plongements de phrases et dans les deux dernières branches, il intègre les étiquettes des EP et les plongements de mots pour les mots appartenant aux EP. Le deuxième système utilise deux branches neuronales, dont la première branche utilise les plongements de phrases. Dans la deuxième branche, le système intègre les plongements de mots et les étiquettes des EP. Ci-dessous, nos conclusions :

— Nous avons constaté que l'intégration des informations des EP aide à la détection des discours haineux pour les deux systèmes proposés comparé à un système n'utilisant pas les EP.

— Nous avons observé que le système à deux branches neuronales obtenait de meilleurs performances que celui fondé sur trois branches. Cela est probablement dû au fait que le système à deux branches utilise tous les plongements de mots de la phrase comparé au système à trois branches qui intègre uniquement les plongements des mots appartenant aux EP.

Dans le **chapitre 7**, nous avons traité notre troisième question de recherche. Nous avons proposé un système fondé sur l'apprentissage multi-tâches avec un mécanisme d'auto-attention

multi-têtes. L'objectif de notre système est d'apprendre à faire attention aux EP pour aider la détection des discours haineux. Notre système utilise des plongements de mots. Nous avons évalué expérimentalement plusieurs plongements contextuels de mots provenant de différents modèles de langages à savoir BERTweet, HateBERT et fBERT. Voici nos conclusions :

— Nous avons observé que notre système utilisant les plongements de BERTweet et de HateBERT surpasse significativement un système de base n'utilisant ni l'auto-attention, ni l'apprentissage multi-tâches.

— Nous avons remarqué que le système proposé obtenait de meilleures performances que le système neuronal à deux branches utilisant les EP.

Ensuite, nous nous sommes concentré sur une nouvelle méthode d'apprentissage peu étudiée pour détecter les tweets haineux. Nous avons exploré l'utilisation de l'apprentissage par contraste qui a pour objectif de rapprocher les représentations de tweets de la même classe dans l'espace vectoriel et de les éloigner pour les tweets de classes différentes.

Dans le **chapitre 8**, nous avons proposé une étude préliminaire de l'utilisation de l'apprentissage par contraste pour la détection des discours de haine. Pour cela, nous avons basé notre approche sur des paires de tweets. Nous avons proposé deux méthodes de création de paires : une fondée sur un tirage aléatoire et l'autre utilisant la similarité cosinus. Le système proposé est inspiré du système SentenceBERT utilisant un modèle siamois (partage des poids). Nous avons également proposé deux stratégies de décision basées sur la similarité cosinus pour la classification des tweets. La première stratégie utilise le vote majoritaire. La deuxième stratégie utilise la moyenne des similarités cosinus. Nous avons tiré les conclusion suivantes :

— Nous avons constaté que la stratégie de prise de décision utilisant la moyenne des similarités cosinus surpasse la stratégie du vote majoritaire.

— Nous avons observé des performances similaires avec les deux méthodes de création de paires de tweets. Cependant, la méthode guidée par la similarité cosinus offre l'avantage d'utiliser moins de paires de tweets.

— Dans cette étude préliminaire, l'apprentissage par contraste n'a pas apporté d'amélioration significative des résultats. Cependant, en utilisant la méthode de projection PCA sur les plongements de phrases, nous avons observé que les classes étaient mieux séparées avec cet apprentissage.

9.2 Perspectives

Dans cette section, nous proposons plusieurs directions de recherche à court et long terme.

9.2.1 Perspectives à court terme

Expressions polylexicales

Une direction de recherche à court terme serait d'explorer la capacité de généralisation de nos approches utilisant les EP. Cela permettrait d'évaluer la robustesse de nos systèmes sur la généralisation entre les jeux de données. En effet, nous pensons que les EP sont importantes pour la détection des discours de haine et permettraient de rendre les systèmes de détection de haine plus robustes pour de nouvelles données.

Nous soulevons aussi une problématique qui est l'inexistence de données annotées en termes d'EP et de haine. Nous soutenons qu'il serait essentiel de développer de tels jeux de données

dans l'avenir. Cela permettrait probablement que nos systèmes fondés sur les EP obtiennent de meilleures performances pour détecter les discours de haine.

Apprentissage par contraste

L'utilisation de l'apprentissage par contraste reste une direction de recherche ouverte. Dans cette thèse, nous avons effectué une étude préliminaire. Une amélioration possible de notre approche serait d'utiliser un classifieur avec les plongements de phrases appris grâce à l'apprentissage par contraste. En effet, nous avons observé que les classes étaient mieux séparées en utilisant l'apprentissage par contraste.

Nous proposons aussi une direction de recherche qui consisterait à étudier l'apprentissage par contraste en utilisant plusieurs jeux de données. Cela permettrait d'étudier de l'apport de cet apprentissage pour la généralisation de la détection des discours de haine. De plus, l'utilisation de l'apprentissage par contraste en mélangeant les données de différents ensembles d'apprentissage permettrait probablement d'atténuer les différents biais contenus sur les jeux de données.

9.2.2 Perspectives à long terme

Apprentissage par contraste

Nous avons effectué dans cette thèse une étude préliminaire de l'apprentissage par contraste. Cependant, l'utilisation de cet apprentissage reste très peu exploré dans le domaine de la détection des discours de haine. Il existe des méthodes de l'apprentissage par contraste, provenant de la vision par ordinateur, qui n'ont pas encore été exploré pour le TAL, comme par exemple *Barlow Twins* (Zbontar et al., 2021). D'autres méthodes d'apprentissage par contraste, comme par exemple celle de Gao et al. (2021), nécessitent de l'augmentation de données pour être appliqué. Cela ouvre un large panorama de recherche concernant l'apprentissage par contraste pour la détection des discours de haine.

Utilisation du contexte des évènements passés, actuels ou futurs

La majorité des jeux de données pour la détection des discours de haine ne contiennent pas d'informations contextuelles, comme le fil de discussion, les messages antérieurs, les évènements, etc. Cependant certains chercheurs soutiennent le fait que des contenus haineux dépendent du contexte dans lequel ils apparaissent. Il serait important de développer des ressources contenant de telles informations. Nous pensons qu'il est important d'étudier comment utiliser ses informations pour détecter les discours de haine.

Utilisation de la multi-modalité

Les tweets peuvent contenir d'autres sources d'informations, telles que des images, vidéos, etc. Un défi de la détection des discours de haine est l'utilisation de la multi-modalité. L'apprentissage multi-modale consiste à apprendre des caractéristiques provenant de sources différentes pour réaliser une tâche. Dans le cas de la détection des discours de haine, l'objectif serait de savoir comment combiner les différentes informations provenant de différentes sources.

Grands modèles de langages

Les récentes avancées dans le traitement automatique des langues, comme les grands modèles de langages (LLM pour *Large Language Model* en anglais), pourraient être utilisées pour détecter

les discours haineux (Møller et al., 2023). En effet, les LLM possèdent énormément de paramètres avec de très grandes capacités de génération de texte et de connaissance du langage. L'utilisation des LLM permettrait d'effectuer de l'apprentissage guidé (*prompting learning* en anglais). Les LLM pourraient être utilisés d'une autre façon, comme par exemple pour augmenter les données (Dai et al., 2023) afin de palier au déséquilibre des classes dans les jeux de données. Cependant, d'un autre point de vue, nous pensons qu'il est important que de tels modèles ne soient pas dans la capacité de générer des propos haineux car ils sont de plus en plus souvent utilisés par la société. Nous soulevons une question qui nous semble importante : comment éviter que les LLM génèrent des contenus haineux ?

Bibliographie

Al Kuwatly, H., Wich, M., and Groh, G. (2020). Identifying and measuring annotator bias based on annotators' demographic characteristics. In *Proceedings of the Fourth Workshop on Online Abuse and Harms*, pages 184–190, Online. Association for Computational Linguistics.

Awal, M. R., Cao, R., Lee, R. K.-W., and Mitrovic, S. (2021a). Angrybert : Joint learning target and emotion for hate speech detection. In *PAKDD*.

Awal, M. R., Cao, R., Lee, R. K.-W., and Mitrović, S. (2021b). Angrybert : Joint learning target and emotion for hate speech detection. In Karlapalem, K., Cheng, H., Ramakrishnan, N., Agrawal, R. K., Reddy, P. K., Srivastava, J., and Chakraborty, T., editors, *Advances in Knowledge Discovery and Data Mining*, pages 701–713, Cham. Springer International Publishing.

Badjatiya, P., Gupta, S., Gupta, M., and Varma, V. (2017). Deep learning for hate speech detection in tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*, WWW '17 Companion, pages 759–760, Republic and Canton of Geneva, CHE.

Badri, N., Kboubi, F., and Chaibi, A. H. (2022). Combining fasttext and glove word embedding for offensive and hate speech text detection. *Procedia Computer Science*, 207 :769–778.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv :1409.0473*.

Baldwin, T. and Kim, S. (2010). Multiword expressions. In *Handbook of Natural Language Processing*, pages 267–292. CRC Press, Taylor and Francis Group, Boca Raton, 2nd edition.

Basile, V., Bosco, C., Fersini, E., Nozza, D., Patti, V., Rangel Pardo, F. M., Rosso, P., and Sanguinetti, M. (2019). SemEval-2019 task 5 : Multilingual detection of hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Basile, V., Croce, D., Maro, M. D., and Passaro, L. C., editors (2020). *Proceedings of the Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020), Online event, December 17th, 2020*, volume 2765 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Berk, G., Erden, B., and Güngör, T. (2018). Deep-BGT at PARSEME shared task 2018 : Bidirectional LSTM-CRF model for verbal multiword expression identification. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 248–253, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Bies, A., Mott, J., Warner, C., and Kulick, S. (2012). English web treebank. *Linguistic Data Consortium, Philadelphia, PA*.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5 :135–146.

Boros, T. and Burtica, R. (2018). GBD-NER at PARSEME shared task 2018 : Multi-word expression detection using bidirectional long-short-term memory networks and graph-based decoding. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 254–260, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Bosco, C., Dell'Orletta, F., Poletto, F., Sanguinetti, M., and Tesconi, M. (2018). Overview of the evalita 2018 hate speech detection task. In *EVALITA Evaluation of NLP and Speech Tools for Italian*, pages 67–74. Accademia University Press.

Bose, T. (2023). *Transfer learning for abusive language detection.* PhD thesis, Université de Lorraine.

Breidt, E., Segond, F., and Valetto, G. (1996). Formal description of multi-word lexemes with the finite-state formalism IDAREX. In *COLING 1996 Volume 2 : The 16th International Conference on Computational Linguistics.*

Breitfeller, L., Ahn, E., Jurgens, D., and Tsvetkov, Y. (2019). Finding microaggressions in the wild : A case for locating elusive phenomena in social media posts. 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)*, pages 1664–1674, Hong Kong, China. Association for Computational Linguistics.

Burnap, P., Rana, O. F., Avis, N., Williams, M., Housley, W., Edwards, A., Morgan, J., and Sloan, L. (2015). Detecting tension in online communities with computational twitter analysis. *Technological Forecasting and Social Change*, 95 :96–108.

Burnap, P. and Williams, M. L. (2015). Cyber hate speech on twitter : An application of machine classification and statistical modeling for policy and decision making. *Policy & Internet*, 7(2) :223–242.

Cai, T. T., Frankle, J., Schwab, D. J., and Morcos, A. S. (2020). Are all negatives created equal in contrastive instance discrimination ? *CoRR*, abs/2010.06682.

Cambria, E., Li, Y., Xing, F. Z., Poria, S., and Kwok, K. (2020). Senticnet 6 : Ensemble application of symbolic and subsymbolic ai for sentiment analysis. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, page 105–114, New York, NY, USA. Association for Computing Machinery.

Carpuat, M. and Diab, M. (2010). Task-based evaluation of multiword expressions : a pilot study in statistical machine translation. In *Human Language Technologies : The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 242–245, Los Angeles, California. Association for Computational Linguistics.

Caselli, T., Basile, V., Mitrović, J., and Granitzer, M. (2021). HateBERT : Retraining BERT for abusive language detection in English. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 17–25, Online. Association for Computational Linguistics.

Caselli, T., Basile, V., Mitrović, J., Kartoziya, I., and Granitzer, M. (2020). I feel offended, don't be abusive ! implicit/explicit messages in offensive and abusive language. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 6193–6202, Marseille, France. European Language Resources Association.

Caselli, T., Novielli, N., Patti, V., and Rosso, P. (2018). Evalita 2018 : Overview on the 6th evaluation campaign of natural language processing and speech tools for italian. In Caselli, T., Novielli, N., Patti, V., and Rosso, P., editors, *Proceedings of the Sixth Evaluation Campaign of*

Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy, December 12-13, 2018, volume 2263 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Cer, D., Yang, Y., Kong, S.-y., Hua, N., Limtiaco, N., St. John, R., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Strope, B., and Kurzweil, R. (2018). Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing : System Demonstrations*, pages 169–174, Brussels, Belgium. Association for Computational Linguistics.

Chakrabarty, T., Gupta, K., and Muresan, S. (2019). Pay "attention" to your context when classifying abusive language. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 70–79, Florence, Italy. Association for Computational Linguistics.

Chen, Q., Zhang, R., Zheng, Y., and Mao, Y. (2022). Dual contrastive learning : Text classification via label-aware data augmentation. *CoRR*, abs/2201.08702.

Chen, Y., Zhou, Y., Zhu, S., and Xu, H. (2012). Detecting offensive language in social media to protect adolescent online safety. In *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, pages 71–80.

Chiril, P., Pamungkas, E. W., Benamara, F., Moriceau, V., and Patti, V. (2022). Emotionally informed hate speech detection : a multi-target perspective. *Cognitive Computation*, pages 1–31.

Cho, K., Van Merriënboer, B., Bahdanau, D., and Bengio, Y. (2014a). On the properties of neural machine translation : Encoder-decoder approaches. *arXiv preprint arXiv :1409.1259*.

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

Conneau, A., Kiela, D., Schwenk, H., Barrault, L., and Bordes, A. (2017). Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

Constant, M., Eryiğit, G., Monti, J., van der Plas, L., Ramisch, C., Rosner, M., and Todirascu, A. (2017). Survey : Multiword expression processing : A Survey. *Computational Linguistics*, 43(4) :837–892.

Constant, M. and Nivre, J. (2016). A transition-based system for joint lexical and syntactic analysis. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 161–171, Berlin, Germany. Association for Computational Linguistics.

Corazza, M., Menini, S., Cabrio, E., Tonelli, S., and Villata, S. (2020). A multilingual evaluation for online hate speech detection. *ACM Trans. Internet Technol.*, 20(2).

Cordeiro, S., Ramisch, C., and Villavicencio, A. (2016). UFRGS&LIF at SemEval-2016 task 10 : Rule-based MWE identification and predominant-supersense tagging. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 910–917, San Diego, California. Association for Computational Linguistics.

Dai, H., Liu, Z., Liao, W., Huang, X., Cao, Y., Wu, Z., Zhao, L., Xu, S., Liu, W., Liu, N., et al. (2023). Auggpt : Leveraging chatgpt for text data augmentation. *arXiv preprint arXiv :2302.13007*.

Davidson, T., Warmsley, D., Macy, M. W., and Weber, I. (2017). Automated hate speech detection and the problem of offensive language. *CoRR*, abs/1703.04009.

del Arco, F. M. P., Halat, S., Padó, S., and Klinger, R. (2021). Multi-task learning with sentiment, emotion, and target detection to recognize hate speech and offensive language. *CoRR*, abs/2109.10255.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT : Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Djuric, N., Zhou, J., Morris, R., Grbovic, M., Radosavljevic, V., and Bhamidipati, N. (2015). Hate speech detection with comment embeddings. In *Proceedings of the 24th international conference on world wide web*, pages 29–30.

D'Sa, A. G., Illina, I., and Fohr, D. (2020). Bert and fasttext embeddings for automatic detection of toxic speech. In *2020 International Multi-Conference on : "Organization of Knowledge and Advanced Technologies" (OCTA)*, pages 1–5.

D'Sa, A. G., Illina, I., Fohr, D., and Akbar, A. (2022). Exploration of multi-corpus learning for hate speech classification in low resource scenarios. In Sojka, P., Horák, A., Kopeček, I., and Pala, K., editors, *Text, Speech, and Dialogue*, pages 238–250, Cham. Springer International Publishing.

Ehren, R., Lichte, T., and Samih, Y. (2018). Mumpitz at PARSEME shared task 2018 : A bidirectional LSTM for the identification of verbal multiword expressions. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 261–267, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Eisner, B., Rocktäschel, T., Augenstein, I., Bošnjak, M., and Riedel, S. (2016). emoji2vec : Learning emoji representations from their description. In *Proceedings of the Fourth International Workshop on Natural Language Processing for Social Media*, pages 48–54, Austin, TX, USA. Association for Computational Linguistics.

ElSherief, M., Ziems, C., Muchlinski, D., Anupindi, V., Seybolt, J., De Choudhury, M., and Yang, D. (2021). Latent hatred : A benchmark for understanding implicit hate speech. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 345–363, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Founta, A., Djouvas, C., Chatzakou, D., Leontiadis, I., Blackburn, J., Stringhini, G., Vakali, A., Sirivianos, M., and Kourtellis, N. (2018). Large scale crowdsourcing and characterization of twitter abusive behavior. *Proceedings of the International AAAI Conference on Web and Social Media*, 12.

Gambäck, B. and Sikdar, U. K. (2017a). Using convolutional neural networks to classify hate-speech. In *Proceedings of the first workshop on abusive language online*, pages 85–90.

Gambäck, B. and Sikdar, U. K. (2017b). Using convolutional neural networks to classify hate-speech. In *Proceedings of the First Workshop on Abusive Language Online*, pages 85–90, Vancouver, BC, Canada. Association for Computational Linguistics.

Gambino, G. and Pirrone, R. (2020). CHILab @ HaSpeeDe 2 : Enhancing hate speech detection with part-of-speech tagging. In *EVALITA Evaluation of NLP and Speech Tools for Italian*, pages 165–170. Accademia University Press.

Gao, T., Yao, X., and Chen, D. (2021). Simcse : Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Ghoneim, M. and Diab, M. (2013). Multiword expressions in the context of statistical machine translation. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1181–1187, Nagoya, Japan. Asian Federation of Natural Language Processing.

Gillick, L. and Cox, S. J. (1989). Some statistical issues in the comparison of speech recognition algorithms. In *International Conference on Acoustics, Speech, and Signal Processing,*, pages 532–535 vol.1.

Gitari, N. D., Zuping, Z., Damien, H., and Long, J. (2015). A lexicon-based approach for hate speech detection. *International Journal of Multimedia and Ubiquitous Engineering*, 10(4) :215–230.

Gross, M. (1989). The use of finite automata in the lexical representation of natural language. In Gross, M. and Perrin, D., editors, *Electronic Dictionaries and Automata in Computational Linguistics*, pages 34–50, Berlin, Heidelberg. Springer Berlin Heidelberg.

Hadsell, R., Chopra, S., and LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. B. (2019). Momentum contrast for unsupervised visual representation learning. *CoRR*, abs/1911.05722.

Hearst, M., Dumais, S., Osuna, E., Platt, J., and Scholkopf, B. (1998). Support vector machines. *IEEE Intelligent Systems and their Applications*, 13(4) :18–28.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8) :1735–1780.

Indurthi, V., Syed, B., Shrivastava, M., Chakravartula, N., Gupta, M., and Varma, V. (2019). FERMI at SemEval-2019 task 5 : Using sentence embeddings to identify hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 70–74, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Jaiswal, A., Babu, A. R., Zadeh, M. Z., Banerjee, D., and Makedon, F. (2020). A survey on contrastive self-supervised learning. *Technologies*, 9(1) :2.

Jurgens, D., Hemphill, L., and Chandrasekharan, E. (2019). A just and comprehensive strategy for using NLP to address online abuse. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3658–3666, Florence, Italy. Association for Computational Linguistics.

Kapil, P. and Ekbal, A. (2020). A deep neural network based multi-task learning approach to hate speech detection. *Knowledge-Based Systems*, 210 :106458.

Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv :1408.5882*.

Kim, Y., Park, S., and Han, Y.-S. (2022). Generalizable implicit hate speech detection using contrastive learning. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 6667–6679, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Klein, D. and Manning, C. D. (2004). *Parsing and Hypergraphs*, pages 351–372. Springer Netherlands, Dordrecht.

Kurfalı, M. (2020). TRAVIS at PARSEME shared task 2020 : How good is (m)BERT at seeing the unseen ? In *Proceedings of the Joint Workshop on Multiword Expressions and Electronic Lexicons*, pages 136–141, online. Association for Computational Linguistics.

Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR.

Lin, Z., Feng, M., Santos, C. N. d., Yu, M., Xiang, B., Zhou, B., and Bengio, Y. (2017). A structured self-attentive sentence embedding. *arXiv preprint arXiv :1703.03130*.

Liu, N. F., Hershcovich, D., Kranzlein, M., and Schneider, N. (2021). Lexical semantic recognition. In *Proceedings of the 17th Workshop on Multiword Expressions (MWE 2021)*, pages 49–56, Online. Association for Computational Linguistics.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019a). Roberta : A robustly optimized bert pretraining approach. *arXiv preprint arXiv :1907.11692*.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019b). Roberta : A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Liu, Y., Zhu, Y., Che, W., Qin, B., Schneider, N., and Smith, N. A. (2018). Parsing tweets into Universal Dependencies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long Papers)*, pages 965–975, New Orleans, Louisiana. Association for Computational Linguistics.

MacAvaney, S., Yao, H.-R., Yang, E., Russell, K., Goharian, N., and Frieder, O. (2019). Hate speech detection : Challenges and solutions. *PLOS ONE*, 14(8) :1–16.

Maisto, A., Pelosi, S., Vietri, S., Vitale, P., and Paolo II, V. G. (2017). Mining offensive language on social media. In *Proceedings of the Fourth Italian Conference on Computational Linguistics CLiC-it*, pages 252–256.

Maldonado, A., Han, L., Moreau, E., Alsulaimani, A., Chowdhury, K. D., Vogel, C., and Liu, Q. (2017). Detection of verbal multi-word expressions via conditional random fields with syntactic dependency features and semantic re-ranking. In *Proceedings of the 13th Workshop on Multiword Expressions (MWE 2017)*, pages 114–120, Valencia, Spain. Association for Computational Linguistics.

Mandl, T., Modha, S., Kumar M, A., and Chakravarthi, B. R. (2021). Overview of the hasoc track at fire 2020 : Hate speech and offensive language identification in tamil, malayalam, hindi, english and german. In *Proceedings of the 12th Annual Meeting of the Forum for Information Retrieval Evaluation*, FIRE '20, page 29–32, New York, NY, USA. Association for Computing Machinery.

McCullagh, P. and Nelder, J. (1989). *Generalized Linear Models, Second Edition*. Chapman and Hall/CRC Monographs on Statistics and Applied Probability Series. Chapman & Hall.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv :1301.3781*.

Møller, A. G., Dalsgaard, J. A., Pera, A., and Aiello, L. M. (2023). Is a prompt and a few samples all you need ? using gpt-4 for data augmentation in low-resource classification tasks. *arXiv preprint arXiv :2304.13861*.

Montariol, S., Riabi, A., and Seddah, D. (2022). Multilingual auxiliary tasks training : Bridging the gap between languages for zero-shot transfer of hate speech detection models. In *Findings of the Association for Computational Linguistics : AACL-IJCNLP 2022*, pages 347–363, Online only. Association for Computational Linguistics.

Mozafari, M., Farahbakhsh, R., and Crespi, N. (2020). A bert-based transfer learning approach for hate speech detection in online social media. In Cherifi, H., Gaito, S., Mendes, J. F., Moro, E., and Rocha, L. M., editors, *Complex Networks and Their Applications VIII*, pages 928–940, Cham. Springer International Publishing.

Narasimhan, H., Pan, W., Kar, P., Protopapas, P., and Ramaswamy, H. G. (2016). Optimizing the multiclass f-measure via biconcave programming. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 1101–1106.

Naseem, U., Razzak, I., and Hameed, I. A. (2019). Deep context-aware embedding for abusive and hate speech detection on twitter. *Aust. J. Intell. Inf. Process. Syst.*, 15(3) :69–76.

Nguyen, D. Q., Vu, T., and Tuan Nguyen, A. (2020). BERTweet : A pre-trained language model for English tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing : System Demonstrations*, pages 9–14, Online. Association for Computational Linguistics.

Njagi, Dennis, G., Zuping, Z., Hanyurwimfura, D., and Long, J. (2015). A lexicon-based approach for hate speech detection. *International Journal of Multimedia and Ubiquitous Engineering*, 10 :215–230.

Nobata, C., Tetreault, J., Thomas, A., Mehdad, Y., and Chang, Y. (2016a). Abusive language detection in online user content. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, page 145–153, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Nobata, C., Tetreault, J., Thomas, A., Mehdad, Y., and Chang, Y. (2016b). Abusive language detection in online user content. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, page 145–153, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Nockeby, J. T. (2000). Hate speech. In Levy, L. W., Karst, K. L., and Mahoney, D. J., editors, *Encyclopedia of the American Constitution*, pages 1277–1279. Macmillan 2nd edition.

Pamungkas, E. W. and Patti, V. (2019). Cross-domain and cross-lingual abusive language detection : A hybrid approach with deep learning and a multilingual lexicon. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics : Student Research Workshop*, pages 363–370, Florence, Italy. Association for Computational Linguistics.

Pappas, N. and Henderson, J. (2019). GILE : A generalized input-label embedding for text classification. *Transactions of the Association for Computational Linguistics*, 7 :139–155.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove : Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Plaza-Del-Arco, F. M., Molina-González, M. D., Ureña-López, L. A., and Martín-Valdivia, M. T. (2021). A multi-task learning approach to hate speech detection leveraging sentiment analysis. *IEEE Access*, 9 :112478–112489.

Poria, S., Gelbukh, A., Hussain, A., Howard, N., Das, D., and Bandyopadhyay, S. (2013). Enhanced senticnet with affective labels for concept-based opinion mining. *IEEE Intelligent Systems*, 28(2) :31–38.

Qi, S., Jin, R., and Paik, J.-Y. (2022). Emoco : Sentence representation learning with enhanced momentum contrast. In *Proceedings of the 5th International Conference on Computer Science and Software Engineering*, CSSE '22, page 159–163, New York, NY, USA. Association for Computing Machinery.

Rajamanickam, S., Mishra, P., Yannakoudakis, H., and Shutova, E. (2020). Joint modelling of emotion and abusive language detection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4270–4279, Online. Association for Computational Linguistics.

Ramisch, C., Cordeiro, S. R., Savary, A., Vincze, V., Barbu Mititelu, V., Bhatia, A., Buljan, M., Candito, M., Gantar, P., Giouli, V., Güngör, T., Hawwari, A., Iñurrieta, U., Kovalevskaitė, J., Krek, S., Lichte, T., Liebeskind, C., Monti, J., Parra Escartín, C., QasemiZadeh, B., Ramisch, R., Schneider, N., Stoyanova, I., Vaidya, A., and Walsh, A. (2018). Edition 1.1 of the PARSEME shared task on automatic identification of verbal multiword expressions. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 222–240, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Ramisch, C., Savary, A., Guillaume, B., Waszczuk, J., Candito, M., Vaidya, A., Barbu Mititelu, V., Bhatia, A., Iñurrieta, U., Giouli, V., Güngör, T., Jiang, M., Lichte, T., Liebeskind, C., Monti, J., Ramisch, R., Stymne, S., Walsh, A., and Xu, H. (2020). Edition 1.2 of the PARSEME shared task on semi-supervised identification of verbal multiword expressions. In *Proceedings of the Joint Workshop on Multiword Expressions and Electronic Lexicons*, pages 107–118, online. Association for Computational Linguistics.

Ramisch, C., Villavicencio, A., and Boitet, C. (2010). mwetoolkit : a framework for multiword expression identification. In *LREC*.

Ramisch, C., Walsh, A., Blanchard, T., and Taslimipoor, S. (2023). A survey of MWE identification experiments : The devil is in the details. In *Proceedings of the 19th Workshop on Multiword Expressions (MWE 2023)*, pages 106–120, Dubrovnik, Croatia. Association for Computational Linguistics.

Razavi, A. H., Inkpen, D., Uritsky, S., and Matwin, S. (2010). Offensive language detection using multi-level classification. In Farzindar, A. and Kešelj, V., editors, *Advances in Artificial Intelligence*, pages 16–27, Berlin, Heidelberg. Springer Berlin Heidelberg.

Razo, D. and Kübler, S. (2020). Investigating sampling bias in abusive language detection. In *Proceedings of the Fourth Workshop on Online Abuse and Harms*, pages 70–78, Online. Association for Computational Linguistics.

Reimers, N. and Gurevych, I. (2019). Sentence-BERT : Sentence embeddings using Siamese BERT-networks. 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)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Rethmeier, N. and Augenstein, I. (2023). A primer on contrastive pretraining in language processing : Methods, lessons learned, and perspectives. *ACM Computing Surveys*, 55(10) :1–17.

Ross, B., Rist, M., Carbonell, G., Cabrera, B., Kurowsky, N., and Wojatzki, M. (2016). Measuring the reliability of hate speech annotations : The case of the european refugee crisis. In Dipper, S., editor, *NLP4CMC III : 3rd Workshop on Natural Language Processing for*

Computer-Mediated Communication, Bochumer Linguistische Arbeitsberichte, pages 6–9, Germany. Ruhr-Universitat Bochum. Originally published in Bochumer Linguistische Arbeitsberichte 17, NLP4CMC III : 3rd Workshop on Natural Language Processing for Computer-Mediated Communication, by Michael Beißwenger, Michael Wojatzki and Torsten Zesch (Eds.), 22 September 2016 (ISSN 2190-0949).; 3rd Workshop on Natural Language Processing for Computer-Mediated Communication / Social Media, NLP4CMC 2016; Conference date : 22-09-2016 Through 22-09-2016.

Rücklé, A., Eger, S., Peyrard, M., and Gurevych, I. (2018). Concatenated power mean word embeddings as universal cross-lingual sentence representations. *arXiv preprint arXiv :1803.01400*.

Safi Samghabadi, N., Patwa, P., PYKL, S., Mukherjee, P., Das, A., and Solorio, T. (2020). Aggression and misogyny detection using BERT : A multi-task approach. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 126–131, Marseille, France. European Language Resources Association (ELRA).

Sag, I. A., Baldwin, T., Bond, F., Copestake, A., and Flickinger, D. (2002). Multiword expressions : A pain in the neck for nlp. In Gelbukh, A., editor, *Computational Linguistics and Intelligent Text Processing*, pages 1–15, Berlin, Heidelberg. Springer Berlin Heidelberg.

Sap, M., Swayamdipta, S., Vianna, L., Zhou, X., Choi, Y., and Smith, N. A. (2022). Annotators with attitudes : How annotator beliefs and identities bias toxic language detection. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 5884–5906, Seattle, United States. Association for Computational Linguistics.

Sarkar, D., Zampieri, M., Ranasinghe, T., and Ororbia, A. (2021). fBERT : A neural transformer for identifying offensive content. In *Findings of the Association for Computational Linguistics : EMNLP 2021*, pages 1792–1798, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Savary, A., Cordeiro, S., and Ramisch, C. (2019). Without lexicons, multiword expression identification will never fly : A position statement. In *Proceedings of the Joint Workshop on Multiword Expressions and WordNet (MWE-WN 2019)*, pages 79–91, Florence, Italy. Association for Computational Linguistics.

Savary, A., Ramisch, C., Cordeiro, S., Sangati, F., Vincze, V., QasemiZadeh, B., Candito, M., Cap, F., Giouli, V., Stoyanova, I., and Doucet, A. (2017). The PARSEME shared task on automatic identification of verbal multiword expressions. In *Proceedings of the 13th Workshop on Multiword Expressions (MWE 2017)*, pages 31–47, Valencia, Spain. Association for Computational Linguistics.

Schneider, N., Hovy, D., Johannsen, A., and Carpuat, M. (2016). SemEval-2016 task 10 : Detecting minimal semantic units and their meanings (DiMSUM). In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 546–559, San Diego, California. Association for Computational Linguistics.

Schneider, N., Onuffer, S., Kazour, N., Danchik, E., Mordowanec, M. T., Conrad, H., and Smith, N. A. (2014). Comprehensive annotation of multiword expressions in a social web corpus. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 455–461, Reykjavik, Iceland. European Language Resources Association (ELRA).

Schneider, N. and Smith, N. A. (2015). A corpus and model integrating multiword expressions and supersenses. In *Proceedings of the 2015 Conference of the North American Chapter of the*

Association for Computational Linguistics : Human Language Technologies, pages 1537–1547, Denver, Colorado. Association for Computational Linguistics.

Shapiro, A., Khalafallah, A., and Torki, M. (2022). AlexU-AIC at Arabic hate speech 2022 : Contrast to classify. In *Proceedinsg of the 5th Workshop on Open-Source Arabic Corpora and Processing Tools with Shared Tasks on Qur'an QA and Fine-Grained Hate Speech Detection*, pages 200–208, Marseille, France. European Language Resources Association.

Sokolova, M. and Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4) :427–437.

Sood, S. O., Churchill, E. F., and Antin, J. (2012). Automatic identification of personal insults on social news sites. *Journal of the American Society for Information Science and Technology*, 63(2) :270–285.

Stanković, R., Mitrović, J., Jokić, D., and Krstev, C. (2020). Multi-word expressions for abusive speech detection in Serbian. In *Proceedings of the Joint Workshop on Multiword Expressions and Electronic Lexicons*, pages 74–84, online. Association for Computational Linguistics.

Stodden, R., QasemiZadeh, B., and Kallmeyer, L. (2018). TRAPACC and TRAPACCS at PARSEME shared task 2018 : Neural transition tagging of verbal multiword expressions. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 268–274, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Taslimipoor, S., Bahaadini, S., and Kochmar, E. (2020). MTLB-STRUCT @parseme 2020 : Capturing unseen multiword expressions using multi-task learning and pre-trained masked language models. In *Proceedings of the Joint Workshop on Multiword Expressions and Electronic Lexicons*, pages 142–148, online. Association for Computational Linguistics.

Taslimipoor, S. and Rohanian, O. (2018). SHOMA at parseme shared task on automatic identification of vmwes : Neural multiword expression tagging with high generalisation. *CoRR*, abs/1809.03056.

Thelwall, M., Buckley, K., Paltoglou, G., Cai, D., and Kappas, A. (2010). Sentiment strength detection in short informal text. *Journal of the American society for information science and technology*, 61(12) :2544–2558.

Uehara, Y., Ishigaki, T., Aoki, K., Noji, H., Goshima, K., Kobayashi, I., Takamura, H., and Miyao, Y. (2020). Learning with contrastive examples for data-to-text generation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2352–2362, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Van Hee, C., Jacobs, G., Emmery, C., Desmet, B., Lefever, E., Verhoeven, B., De Pauw, G., Daelemans, W., and Hoste, V. (2018). Automatic detection of cyberbullying in social media text. *PloS one*, 13(10) :e0203794.

Van Hee, C., Lefever, E., Verhoeven, B., Mennes, J., Desmet, B., De Pauw, G., Daelemans, W., and Hoste, V. (2015). Detection and fine-grained classification of cyberbullying events. In *Proceedings of the international conference recent advances in natural language processing*, pages 672–680.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017a). Attention is all you need. *CoRR*, abs/1706.03762.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017b). Attention is all you need. *CoRR*, abs/1706.03762.

Waseem, Z. (2016). Are you a racist or am I seeing things ? annotator influence on hate speech detection on Twitter. In *Proceedings of the First Workshop on NLP and Computational Social Science*, pages 138–142, Austin, Texas. Association for Computational Linguistics.

Waseem, Z. and Hovy, D. (2016). Hateful symbols or hateful people ? predictive features for hate speech detection on Twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.

Waseem, Z., Lulz, S., Bingel, J., and Augenstein, I. (2021). Disembodied machine learning : On the illusion of objectivity in NLP. *CoRR*, abs/2101.11974.

Waseem, Z., Thorne, J., and Bingel, J. (2018). Bridging the gaps : Multi task learning for domain transfer of hate speech detection. *Online harassment*, pages 29–55.

Waszczuk, J. (2018). TRAVERSAL at PARSEME shared task 2018 : Identification of verbal multiword expressions using a discriminative tree-structured model. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 275–282, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Wich, M., Al Kuwatly, H., and Groh, G. (2020). Investigating annotator bias with a graph-based approach. In *Proceedings of the Fourth Workshop on Online Abuse and Harms*, pages 191–199, Online. Association for Computational Linguistics.

Wiegand, M., Ruppenhofer, J., and Kleinbauer, T. (2019). Detection of Abusive Language : the Problem of Biased Datasets. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 602–608, Minneapolis, Minnesota. Association for Computational Linguistics.

Witten, I. H. and Frank, E. (2002). Data mining : Practical machine learning tools and techniques with java implementations. *SIGMOD Rec.*, 31(1) :76–77.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., and Hovy, E. (2016). Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 1480–1489, San Diego, California. Association for Computational Linguistics.

Yin, W., Schütze, H., Xiang, B., and Zhou, B. (2016). Abcnn : Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for computational linguistics*, 4 :259–272.

Yin, W. and Zubiaga, A. (2021). Towards generalisable hate speech detection : a review on obstacles and solutions. *CoRR*, abs/2102.08886.

Zampieri, N., Illina, I., and Fohr, D. (2021). Multiword expression features for automatic hate speech detection. In Métais, E., Meziane, F., Horacek, H., and Kapetanios, E., editors, *Natural Language Processing and Information Systems*, pages 156–164, Cham. Springer International Publishing.

Zampieri, N., Illina, I., and Fohr, D. (2023). Improving hate speech detection with self-attention mechanism and multi-task learning. In *LTC'23-10th Language & Technology Conference : Human Language Technologies as a Challenge for Computer Science and Linguistics*.

Zampieri, N., Ramisch, C., Illina, I., and Fohr, D. (2022a). Identification des expressions polylexicales dans les tweets (identification of multiword expressions in tweets). In *Actes de la 29e Conférence sur le Traitement Automatique des Langues Naturelles. Volume 1 : conférence principale*, pages 365–373, Avignon, France. ATALA.

Zampieri, N., Ramisch, C., Illina, I., and Fohr, D. (2022b). Identification of multiword expressions in tweets for hate speech detection. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 202–210, Marseille, France. European Language Resources Association.

Zampieri, N., Scholivet, M., Ramisch, C., and Favre, B. (2018). Veyn at PARSEME shared task 2018 : Recurrent neural networks for VMWE identification. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 290–296, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Zbontar, J., Jing, L., Misra, I., LeCun, Y., and Deny, S. (2021). Barlow twins : Self-supervised learning via redundancy reduction. *CoRR*, abs/2103.03230.

Zhang, R., Ji, Y., Zhang, Y., and Passonneau, R. J. (2022). Contrastive data and learning for natural language processing. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies : Tutorial Abstracts*, pages 39–47, Seattle, United States. Association for Computational Linguistics.

Zhang, Z., Robinson, D., and Tepper, J. (2018). Detecting hate speech on twitter using a convolution-gru based deep neural network. In *The Semantic Web : 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15*, pages 745–760. Springer.

Zhao, Q., Xiao, Y., and Long, Y. (2021). Multi-task cnn for abusive language detection. In *2021 IEEE 2nd International Conference on Pattern Recognition and Machine Learning (PRML)*, pages 286–291.

Zhong, W., Wu, Q., Lu, G., Xue, Y., and Hu, X. (2022). Keyword-enhanced multi-expert framework for hate speech detection. *Mathematics*, 10(24).

Zhou, X., Yong, Y., Fan, X., Ren, G., Song, Y., Diao, Y., Yang, L., and Lin, H. (2021). Hate speech detection based on sentiment knowledge sharing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1 : Long Papers)*, pages 7158–7166, Online. Association for Computational Linguistics.