Classification automatique de textes par réseaux de neurones profonds : application au domaine de la santé

Yves Mercadier

THÈSE POUR OBTENIR LE GRADE DE DOCTEUR DE L'UNIVERSITÉ DE MONTPELLIER

En informatique

École doctorale Information, Structures, Systèmes

Unité de recherche LIRMM

Classification automatique de textes par réseaux de neurones profonds : Application au domaine de la santé

Présentée par Yves Mercadier
Le 17 novembre 2020

Sous la direction de Sandra Bringay et Jérôme Azé

Rapporteurs

Mme Sylvie Despres, Professeur des universités, Université Sorbonne Paris Nord
Mme Catherine Faron, Maître de Conférences, Université Côte d'Azur

Devant le jury composé de

Mme Fleur Mougin, Maître de conférences, Université de Bordeaux	Examinatrice
Mme Nathalie Pernelle, Professeur des universités, Université Paris Nord	Examinatrice
Mme Maguelonne Teisseire, Directrice de Recherche, Maison de la Télédetection, Montpellier	Présidente de jury

UNIVERSITÉ DE MONTPELLIER

Résumé

Cette thèse porte sur l'analyse de données textuelles dans le domaine de la santé et en particulier sur la classification supervisée multi-classes de données issues de la littérature biomédicale et des médias sociaux.

Une des difficultés majeures lors de l'exploration de telles données par des méthodes d'apprentissage supervisées est de posséder un jeu de données suffisant en nombre d'exemples pour l'entraînement des modèles. En effet, il est généralement nécessaire de catégoriser les données manuellement avant de réaliser l'étape d'apprentissage. La taille importante des jeux de données rend cette tâche de catégorisation très coûteuse, qu'il convient de réduire par des systèmes semi-automatiques.

Dans ce contexte, l'apprentissage actif, pendant lequel l'oracle intervient pour choisir les meilleurs exemples à étiqueter, s'avère prometteur. L'intuition est la suivante : en choisissant les exemples intelligemment et non aléatoirement, les modèles devraient s'améliorer avec moins d'efforts pour l'oracle et donc à moindre coût (c'est-a-dire avec moins d'exemples annotés). Dans cette thèse, nous évaluerons différentes approches d'apprentissage actif combinées avec des modèles d'apprentissage profond récents.

Par ailleurs, lorsque l'on dispose de peu de données annotées, une possibilité d'amélioration est d'augmenter artificiellement la quantité de données pendant la phase d'entraînement du modèle, en créant de nouvelles données de manière automatique à partir des données existantes. Plus précisément, il s'agit d'injecter de la connaissance en tenant compte des propriétés invariantes des données par rapport à certaines transformations. Les données augmentées peuvent ainsi couvrir un espace d'entrée inexploré, éviter le sur-apprentissage et améliorer la généralisation du modèle. Dans cette thèse, nous proposerons et évaluerons une nouvelle approche d'augmentation de données textuelles.

Ces deux contributions seront évaluées sur différents jeux de données textuels du domaine de la santé.

This Ph.D focuses on the analysis of textual data in the health domain and in particular on the supervised multi-class classification of data from biomedical literature and social media.

One of the major difficulties when exploring such data by supervised learning methods is to have a sufficient number of data sets for models training. Indeed, it is generally necessary to label manually the data before performing the learning step. The large size of the data sets makes this labellisation task very expensive, which should be reduced with semi-automatic systems.

In this context, active learning, in which the Oracle intervenes to choose the best examples to label, is promising. The intuition is as follows : by choosing the smartly the examples and not randomly, the models should improve with less effort for the oracle and therefore at lower cost (i.e. with less annotated examples). In this PhD, we will evaluate different active learning approaches combined with recent deep learning models.

In addition, when small annotated data set is available, one possibility of improvement is to artificially increase the data quantity during the training phase, by automatically creating new data from existing data. More precisely, we inject knowledge by taking into account the invariant properties of the data with respect to certain transformations. The augmented data can thus cover an unexplored input space, avoid overfitting and improve the generalization of the model. In this Ph.D, we will propose and evaluate a new approach for textual data augmentation.

These two contributions will be evaluated on different textual datasets in the medical domain.

Remerciements

Le domaine de la classification de textes est, par nature, à l'intersection du langage, de l'écriture, de l'informatique, des mathématiques, de la taxonomie. Je tiens par conséquent à exprimer ma profonde reconnaissance aux Professeurs Sandra Bringay et Jérôme Azé pour avoir eu l'audace d'accepter de lancer sur le sujet un physicien de formation ! Qu'ils trouvent ici l'expression de ma gratitude pour la confiance accordée ainsi que pour l'ensemble des conseils avertis qu'ils ont pu me fournir tout au long de ma thèse.

Ce travail a été réalisé au sein de l'école doctorale I2S. Je les remercie pour leur soutien. Je tiens à remercier également Catherine Faron et Sylvie Despres d'avoir bien voulu accepter d'être les rapportrices de mes travaux ainsi que Fleur Mougin pour l'intérêt porté à ce travail. Je les remercie chaleureusement pour les relectures qu'elles ont effectuées, et les discussions constructives dont elles m'ont fait profiter : leur participation à ce jury est pour moi un véritable honneur.

Merci également à tous les membres de l'équipe ADVANSE du LIRMM et particulièrement à Pascal Poncelet pour l'agréable atmosphère de travail et l'ambiance cordiale qu'ils ont su faire régner autour de moi durant ces quelques années, malgré l'intermittence de mes visites ! Merci à ma compagne Sandrine Lefebvre pour son soutien au quotidien. Enfin, je voudrais remercier tout spécialement mon père Albert Mercadier, sans qui cette thèse n'aurait probablement jamais débutée.

Á ma mère,

pour sa merveilleuse tendresse
et sa présence affectueuse
tout au long de ma vie tumultueuse.

La véritable éducation consiste à
pousser les gens à penser
par eux-même.

Noam Chomsky

Table des matières

Liste des figures

Liste des tables

Liste des exemples

Liste des équations

I

Introduction

Plan du chapitre

Aujourd'hui, la médecine moderne est devenue inconcevable sans l'utilisation des données de santé, volumineuses et hétérogènes, issues de la relation patient-médecin (dossiers des patients, résultats de biologie et d'imagerie, de la e-santé, de la télé-médecine, de l'ensemble des NBIC (nanotechnologies, biotechnologies, informatique et sciences cognitives), de la littérature médicale et de l'expression personnelle dans les médias sociaux, etc. Par exemple en France, on recense les données médico-économiques le Sniiram (Système National d'Information Inter-Régimes de l'Assurance Maladie) et ses 8,9 milliards de feuilles de soins, ainsi que les images des 80 millions d'actes d'imagerie effectués chaque année, etc. Dans cette thèse, nous allons nous focaliser sur un type de données de santé particulières, les données textuelles. Dans lesquels, nous retrouvons les textes produits par les patients et les données issues de la littérature biomédicale.

1 Motivations de la recherche

Nous considérerons tout d'abord la littérature biomédicale qui résume les connaissances scientifiques actuelles. La littérature biomédicale est très vaste (plusieurs millions de publications scientifiques) et continue de croître rapidement (plus d'un million par an) via des sites comme PubMed [1], Google Scholar [2], etc. Le processus de recherche documentaire, qui consiste à chercher des informations pertinentes dans cette littérature, pour trouver la réponse à une question spécifique, ou encore pour identifier les auteurs d'articles les plus influents sur un sujet, devient de plus en plus complexe. Convertir cette masse d'information sous une forme structurée est un enjeu majeur et constitue le point de départ du développement et de la mise au point d'outils d'interrogation et de traitement automatique adaptés [Chua and Zhang, 2020].

Nous nous intéresserons également aux médias sociaux en ligne, spécialisés dans un domaine comme les forums de discussions, les tweets, etc. Ces médias sociaux ont de multiples fonctions et applications. Dans un premier temps, ils sont une source d'informations précieuses pour les internautes [Barrington et al., 2012]. Ils sont également utilisés à des fins d'information du grand public par les institutions, dans l'intention de sensibiliser les populations et de réaliser des actes de prévention sur les pratiques à risque. Enfin, ils ont également le rôle d'information et de veille auprès des professionnels de santé afin de parfaire leurs connaissances sur les comportements des patients. Proposer des modèles d'analyses automatiques et efficaces, utilisables par les professionnels de santé à partir de ces ensembles de textes produits par les patients, représente un défi

1. https://pubmed.ncbi.nlm.nih.gov/
2. https://scholar.google.com/

majeur [Zhao et al., 2019].

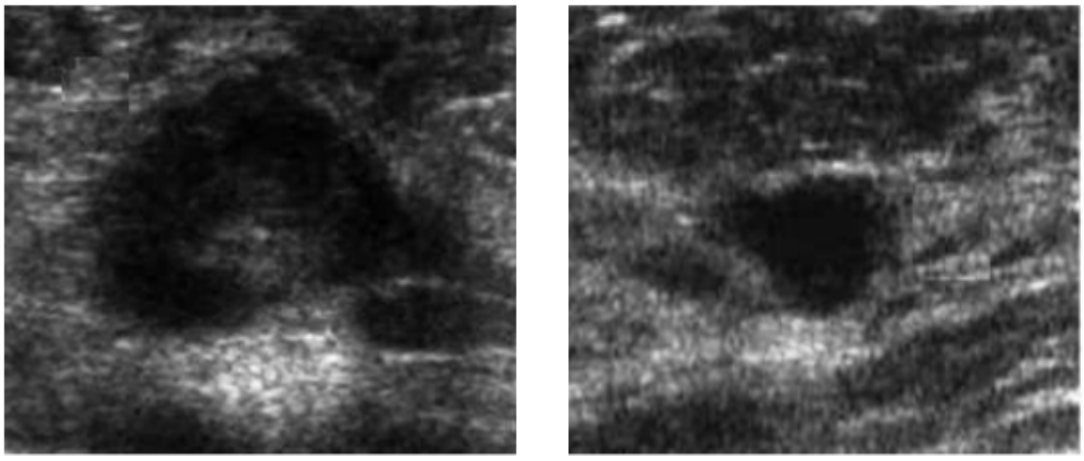
Lorsque l'on veut analyser ces deux types de documents de manière automatique, les méthodes de classification sont essentielles. La classification est une méthode d'analyse de données qui consiste, pour faciliter l'étude d'une population, à regrouper ces individus en classes, de telle sorte que les individus d'une même classe soient les plus semblables possibles et que les individus de classes différentes soient les plus différents possibles. On distingue les approches supervisées pour lesquelles les classes sont connues, des approches non supervisées pour lesquelles les classes ne sont pas connues. Pour comparer les individus, on s'intéresse à des variables correspondant aux caractéristiques de ces individus (e.g. température, vocabulaire, couleur, etc.). L'ensemble de ces variables réalisent une représentation des individus. À partir de ces représentations abstraites, il est alors possible de procéder à la mesure d'une distance entre deux individus. Il existe de nombreux algorithmes de classification qui différent selon le type de données traitées, des choix de représentation de ces données et des modèles sous-jacents.

En santé, parmi les approches de classification ayant connu un vif succès, on peut citer les travaux de [Cheikhrouhou, 2012] qui a travaillé sur la classification des masses mammaires pour le diagnostic du cancer du sein à partir d'imageries (voir figure I.1). Ce type de systèmes automatiques de Diagnostic Assisté par Ordinateur sert à donner un second avis au radiologue. Nous pouvons également citer les travaux de [Robert et al., 2019] pour le Centre d'épidémiologie sur les causes médicales de décès (CépiDc) de l'Inserm qui permet d'identifier automatiquement les causes médicales de décès à partir de textes.

Récemment les modèles de classification par apprentissage profond ont fortement progressé et ont montré leur supériorité sur les modèles de classification plus classiques, notamment sur de très gros ensembles de données. Ils ont été développés et améliorés par les GAFAM[3] qui utilisent des ensembles de données produits par leurs utilisateurs et en conséquence de très grande taille. Or, dans le domaine applicatif de la santé, très fréquemment, les classifications de données utiles se réalisent sur des petits volumes de données textuelles et surtout les modèles sont appris à partir de petits volumes de données annotées. En effet, il est difficile de disposer d'un volume de données étiquetées suffisant pour l'entraînement des modèles. Cette étape est généralement manuelle et coûteuse et il convient de la réduire par des systèmes semi-automatiques.

3. GAFAM : acronyme des géants du web

FIGURE I.1 – Exemples d'images échographiques du sein : a) lésion maligne (sein droit) et b) lésion bénigne (sein gauche de la même patiente) [Cheikhrouhou, 2012]



2 Questions de recherche étudiées et challenges

La question de recherche principale posée dans cette thèse peut être résumée de la manière suivante :

Comment pouvons-nous classer de manière supervisée et efficacement des jeux de données textuels pour lesquels on dispose de peu d'exemples annotés ?

Pour répondre à cette question, nous nous sommes intéressés à trois problèmes sous-jacents.

2.1 Représenter et classer des données textuelles

Pour une tâche précise de classification de données textuelles, le data scientist cherche à identifier le meilleur algorithme de classification et à définir les meilleures caractéristiques prises en entrée de ces classifieurs pour représenter les données. Dernièrement, les méthodes d'apprentissage profond se sont révélées très efficaces pour différents types de données dont les données de nature textuelle. Par ailleurs, les modèles de représentation des textes ont également beaucoup évolué. Les représentations classiques sont généralement basées sur des caractéristiques issues de l'expérience du data scientist (e.g. longueur de la phrase, nombre de chiffres, etc.) et sur des lexiques [Wu et al., 2008]. Ce type d'approches est difficilement généralisable à différents domaines. Récemment,

de nombreux travaux se sont penchés sur les représentations continues de mots (word embeddings) [Mikolov et al., 2013b] et leurs différentes extensions. Nous pouvons généraliser ces aspects sous la forme de la question suivante qui sera traitée dans le chapitre III :

Q1 : Comment pouvons-nous représenter les textes en entrée des algorithmes de classification en évitant les étapes d'ingénierie des caractéristiques spécifiques au domaine et choisir l'algorithme d'apprentissage le plus adapté ?

2.2 Impliquer l'annotateur via un apprentissage actif

En classification de textes, la phase d'étiquetage nécessaire à l'apprentissage du classifieur peut s'avérer longue et fastidieuse. Dans ce contexte, l'apprentissage actif, pendant lequel l'oracle intervient pour choisir les meilleurs exemples à étiqueter, s'avère prometteur. L'intuition est la suivante : en choisissant les exemples intelligemment et non aléatoirement, les modèles devraient s'améliorer avec moins d'efforts pour l'oracle et donc à moindre coût (c'est-a-dire avec moins d'exemples annotés). Il existe différentes approches d'apprentissage actif mais peu ont été combinées avec des modèles d'apprentissage profond [Ash et al., 2020] et encore moins pour des données textuelles [Bang et al., 2018]. Cette problématique peut être résumée de la manière suivante et sera traitée dans le chapitre IV :

Q2 : Comment pouvons-nous mettre en place un processus d'apprentissage actif combiné à un modèle d'apprentissage profond pour choisir les exemples qui vont améliorer le plus rapidement le modèle ?

2.3 Augmenter les données

Lorsque l'on dispose de peu de données annotées, une possibilité d'amélioration est d'augmenter artificiellement la quantité de données pendant la phase d'entraînement du modèle, en créant de nouvelles données de manière automatique à partir des données existantes. Plus précisément, il s'agit d'injecter de la connaissance en tenant compte des propriétés invariantes des données par rapport à certaines transformations. Les données augmentées peuvent ainsi couvrir un espace d'entrée inexploré, éviter le sur-apprentissage et améliorer la généralisation du modèle. Cette technique s'est avérée très efficace pour des tâches de classification d'images [He et al., 2015], en appliquant des changements mineurs sur les données initiales comme des changements d'échelle, des recadrages, des déformations, des rotations, etc. qui ne modifient pas les étiquettes des données car ces changements sont susceptibles de se produire dans des observations du monde réel. Cependant, les transformations qui préservent

les étiquettes pour les données textuelles ne sont pas aussi évidentes et intuitives et peu d'approches [Kobayashi, 2018] ont été proposées. Cette problématique peut être résumée de la manière suivante et sera traitée dans le chapitre V :

Q3 : Comment pouvons-nous mettre en place un processus d'augmentation de données textuelles qui ne modifie pas la distribution des données initiales et qui améliore le modèle d'apprentissage ?

3 Données de la thèse

Dans l'intention de montrer la généricité des approches présentées dans cette thèse, nous avons choisi six jeux de données décrits dans le tableau I.1. Cinq sont issus du domaine de la santé. Deux concernent des documents issus de la littérature médicale. Trois concernent des textes écrits par des patients. Un autre jeu de données a été introduit pour la comparaison à un domaine autre que la santé. Ce jeu est très souvent utilisé pour la comparaison de classifieurs dans la littérature.

Jeu de données	Type	Nombre de documents	Longueur moyenne en nombre de mots	Tâche de classification	Classes
PubMed 200k RCT	article scientifique	2 211 861	26.22	analyse de catégories de textes	5
WHO COVID-19	article scientifique	26 909	166	analyse de provenance	4
eR anorexie	forum	84 834	38.24	analyse de sentiments	2
eR dépression	forum	531 394	36.76	analyse de sentiments	2
Drugs.com	avis	53 766	85.58	analyse de sentiments	10
SST	avis	239232	7.75	analyse de sentiments	5

TABLE I.1 – Jeux de données utilisés dans nos expériences pour la classification de textes.

PubMed 200k RCT[4] [Dernoncourt and Lee, 2017] est une base contenant environ 200 000 résumés d'articles portant sur des essais contrôlés randomisés, totalisant plus de 2 millions de phrases. Chaque phrase est étiquetée selon sa signification dans le résumé (contexte, objectif, méthode, résultat ou conclusion).

WHO COVID-19[5], en réponse à la pandémie de COVID-19, l'institut Allen for AI s'est associé à des groupes de recherche de premier plan pour distribuer un jeu de données composé de plus de 29 000 articles, dont plus de 13 000 textes dans leur intégralité, sur le COVID-19 et la famille des coronavirus. Nous avons utilisé dans cette étude uniquement les articles ayant un résumé. Chaque texte est étiqueté selon sa source : CZI, PMC, biorxiv, medrxiv.

Drugs.com[6] [Gräßer et al., 2018] correspond à des avis de patients sur des médicaments, associés à des maladies. Les données ont été obtenues en analysant les sites pharmaceutiques en ligne. Chaque texte est étiqueté selon une note de 1 à 10 correspondant à la satisfaction des patients.

Les jeux de données eR Depression et eR Anorexie sont issus du challenge CLEF eRisk 2018[7]. Les textes correspondent à des messages d'utilisateurs dans le réseau social [Ragheb et al., 2018]. Chaque texte est étiqueté selon les classes dépression/non dépression et anorexie/non anorexie.

Le jeu de données SST[8] décrit dans [Socher et al., 2013] correspond à des avis d'internautes sur des films. Les documents sont étiquetés de très positif à très négatif sur cinq niveaux.

Une exploration de ces jeux est présentée dans l'annexe A.

4 Résumé des trois contributions principales de la thèse

La thèse aborde les trois questions identifiées dans la section 2 et introduit des solutions pragmatiques à la classification supervisée des textes, quand peu de données sont disponibles, comme cela est souvent le cas dans le domaine de la santé, à partir des données présentées dans la section 3.

Comparaison des classifieurs de textes : Si beaucoup d'approches ont été proposées pour classer des données de nature textuelle dans la littérature, peu de travaux ont comparé de manière exhaustive la représentation des documents et les différents algorithmes d'apprentissage, en particulier, les

4. https://github.com/Franck-Dernoncourt/pubmed-rct
5. https://pages.semanticscholar.org/coronavirus-research
6. https://archive.ics.uci.edu/ml/datasets/Drug+Review+Dataset+%28Drugs.com%29
7. https://early.irlab.org/2018/index.html
8. https://nlp.stanford.edu/sentiment/index.html

dernières architectures d'apprentissage profond. C'est ce que nous nous proposons de faire dans le chapitre III.

Apprentissage profond actif : Guider l'oracle vers les exemples les plus pertinents à annoter pour améliorer les performances du modèle d'apprentissage est une solution particulièrement intéressante dans le domaine de la santé, où les ressources sont limitées. Dans le chapitre IV, nous avons donc adapté les techniques récentes d'apprentissage actif profond utilisées pour la classification d'images, au cas de l'analyse de textes.

Augmentation de données : Nous proposons une nouvelle approche pour augmenter les données textuelles que nous avons appliquée aux jeux de données de la thèse et que nous avons comparée avec succès avec les principales approches de la littérature dans le chapitre V.

5 Plan de la thèse

La figure I.2 montre l'organisation des chapitres de cette thèse. Les chapitres III, IV et V font référence à des articles publiés dans le cadre de cette thèse. La liste de ces articles est présentée dans la section 6 suivante. L'ensemble du travail est organisé comme suit :

— Dans le chapitre II, intitulé « Architecture des réseaux de neurones » , nous décrivons en détail les réseaux de neurones mis en œuvre dans cette thèse à savoir les perceptrons, les réseaux récurrents, les réseaux LSTM, les réseaux de convolution, les auto-encodeurs, le mécanisme d'attention, ainsi que l'architecture transformers.

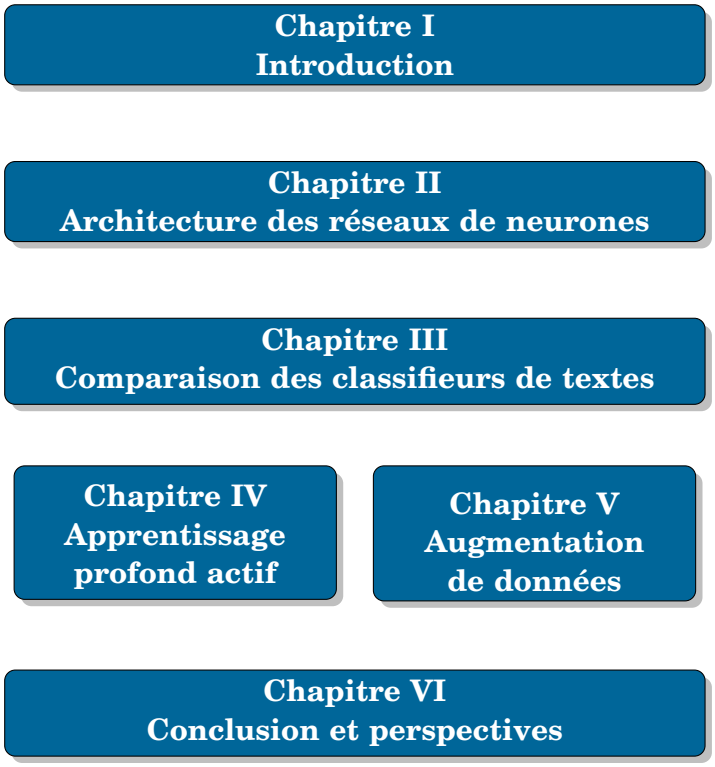
— Dans le chapitre III, intitulé « Comparaison des classifieurs de textes » , nous présentons un état de l'art des algorithmes de classification automatique de textes, puis nous présentons le protocole et les méthodes d'évaluation utilisées dans nos expérimentations sur les jeux de données de la thèse, pour comparer ces différentes architectures.

— Dans le chapitre IV, intitulé « Apprentissage profond actif » , nous nous interrogeons sur l'intérêt de combiner un apprentissage actif avec des architectures profondes. Après un état de l'art des différentes approches d'apprentissage actif, nous expérimentons sur les jeux de la thèse les différentes combinaisons proposées.

— Dans le chapitre V, intitulé « Augmentation de données » , nous présentons un état de l'art sur les méthodes classiques d'augmentation de données puis nous proposons une nouvelle approche que nous validons sur les jeux de données de la thèse et que nous comparons aux approches classiques.

— Le chapitre VI est dédié aux conclusions et perspectives de ces travaux.

FIGURE I.2 – Plan de la thèse



6 Publications

Ce travail de thèse a mené aux publications ci-dessous :

Acte de conférence (IC 2018) : Yves Mercadier, Jérôme Azé, Sandra Bringay, Viviane Clavier, Erick Cuenca, Céline Paganelli, Pascal Poncelet, and Arnaud Sallaberry. #aids analyse information dangers sexualité : caractériser les discours à propos du VIH dans les forums de santé. In Sylvie Ranwez, editor, *IC 2018 : 29es Journées francophones d'Ingénierie des Connaissances (Proceedings of the 29th French Knowledge Engineering Conference), Nancy, France, July 4-6, 2018*, pages 71–86, 2018. URL https://hal.archives-ouvertes.fr/IC_2018/hal-01839552
Article sélectionné parmi les trois « best paper » de la conférence.

Chapitre de livre (Social Web and Health Research) : Yves Mercadier, Bilel Moulahi, Sandra Bringay, Jérôme Azé, Philippe Lenoir, Grégoire Mercier, and François Carbonnel. *Analysis by Multiclass Multilabel Classification of the 2015 #SmearForSmear Campaign Using Deep Learning*, pages 193–205. Springer International Publishing, Cham, 2019. ISBN 978-3-030-14714-3. URL https://doi.org/10.1007/978-3-030-14714-3_10

6.1 Apprentissage actif profond

Post-actes de conférence (EGC2020) : Yves Mercadier, Jérôme Azé et Sandra Bringay. Deep active learning for multi-class text classification. Post-actes de la conférence EGC2020 étendus et en anglais.

À paraître.

Acte de conférence (EGC 2020) : Yves Mercadier, Jérôme Azé, and Sandra Bringay. Apprentissage actif profond pour le classement de textes en plusieurs classes. In Antoine Cornuéjols and Etienne Cuvelier, editors, *Extraction et Gestion des Connaissances, EGC 2020, Brussels, Belgium, January 27-31, 2020*, volume E-36 of *RNTI*, pages 49–60. Éditions RNTI, 2020a. URL http://editions-rnti.fr/?inprocid=1002561

6.2 Augmentation de données

Chapitre de livre (Artificial Intelligence in Medicine) : Yves Mercadier, Sandra Bringay, and Jérôme Azé. *Divide to better classify*, chapter 9. Springer Nature, 7th Floor Block - C Hardy Tower Ramanujan IT City Rajiv Gandhi Salai (OMR) Taramani Chennai - 600 113 India, 2021

À paraître.

Acte de conférence (AIME 2020) : Yves Mercadier, Jérôme Azé, and Sandra Bringay. Divide to better classify. In Martin Michalowski and Robert Moskovitch, editors, *Artificial Intelligence in Medicine - 18th International Conference on Artificial Intelligence in Medicine, AIME 2020, Minneapolis, MN, USA, August 25-28, 2020, Proceedings*, volume 12299 of *Lecture Notes in Computer Science*, pages 89–99. Springer, 2020b. doi: 10.1007/978-3-030-59137-3_9. URL https://doi.org/10.1007/978-3-030-59137-3_9

Article sélectionné parmi les trois « best paper » de la conférence.

Atelier (IAS 2020) : Yves Mercadier, Jérôme Azé et Sandra Bringay. Diviser pour mieux classifier. Atelier IAS 2020 Intelligence Artificielle et Santé associé à la plateforme PFIA 2020, Anger, Juillet 2020

Accepté.

II

Architecture des réseaux de neurones

Plan du chapitre

Dans cette thèse, les méthodes d'apprentissage à base de réseaux de neurones ont constitué la brique de base et font donc l'objet de ce chapitre introductif. Les réseaux de neurones ont aujourd'hui montré de véritables aptitudes dans de nombreuses tâches comme la classification d'images ou de textes, la prédiction de cours de bourse, etc. Intuitivement les réseaux de neurones sont construits par mimétisme avec les fonctions cérébrales du vivant (voir figure II.1). Cette métaphore du vivant ne prétend pas reproduire le fonctionnement biologique du neurone mais plutôt en être une inspiration. On peut citer Yann Lecun à ce sujet « C'est un abus de langage que de parler de neurones ! De la même façon qu'on parle d'aile pour un avion mais aussi pour un oiseau, le neurone artificiel est un modèle extrêmement simplifié de la réalité biologique. » [1].

Nous allons maintenant décrire plus en détail les réseaux de neurones mis en œuvre dans cette thèse à savoir le neurone formel (section 1), les perceptrons (section 2 et 3), les réseaux récurrents (section 4), les réseaux LSTM (section 5), les réseaux de convolution (section 6), les auto-encodeurs (section 7), le mécanisme d'attention (section 8) ainsi que l'architecture transformer (section 9).

1 Le neurone formel

Le premier neuronne formel (voir figure II.2) a été proposé par les neurologues Warren McCulloch et Walter Pitts (What the frog's eye tells the frog's brain) [Lettvin et al., 1959]. Le neurone formel est construit comme un système composé d'une fonction de transfert qui module ses sorties en fonction de ses entrées selon des règles précises, ce qui lui procure l'aptitude à résoudre des problèmes simples. À chaque entrée est associé un poids synaptique. Il calcule tout d'abord la somme pondérée de ses entrées. Le résultat est ensuite transformé par une fonction d'activation non linéaire, la fonction de Heaviside ou fonction marche.

Équation : Neurone formel étape intermédiaire

$$z = \sum_{j=1}^{n} w_j . x_j \tag{II.1}$$

avec $x = (x_1, x_2, ..., x_n)$ vecteur d'entrée, $w = (w_1, w_2, ..., w_n)$ vecteur des poids synaptiques, et w_0 une constante représentant un biais. Le biais procurant une certaine malléabilité de la frontière de décision du neurone.

1. https://www.sciencesetavenir.fr/high-tech/intelligence-artificielle/selon-yann-lecun-l-intelligence-artificielle-a-20-ans-pour-faire-ses-preuves_120121
2. Source : https://commons.wikimedia.org/wiki/File:Neuron-figure-fr.svg?

FIGURE II.1 – Le neurone biologique [2]

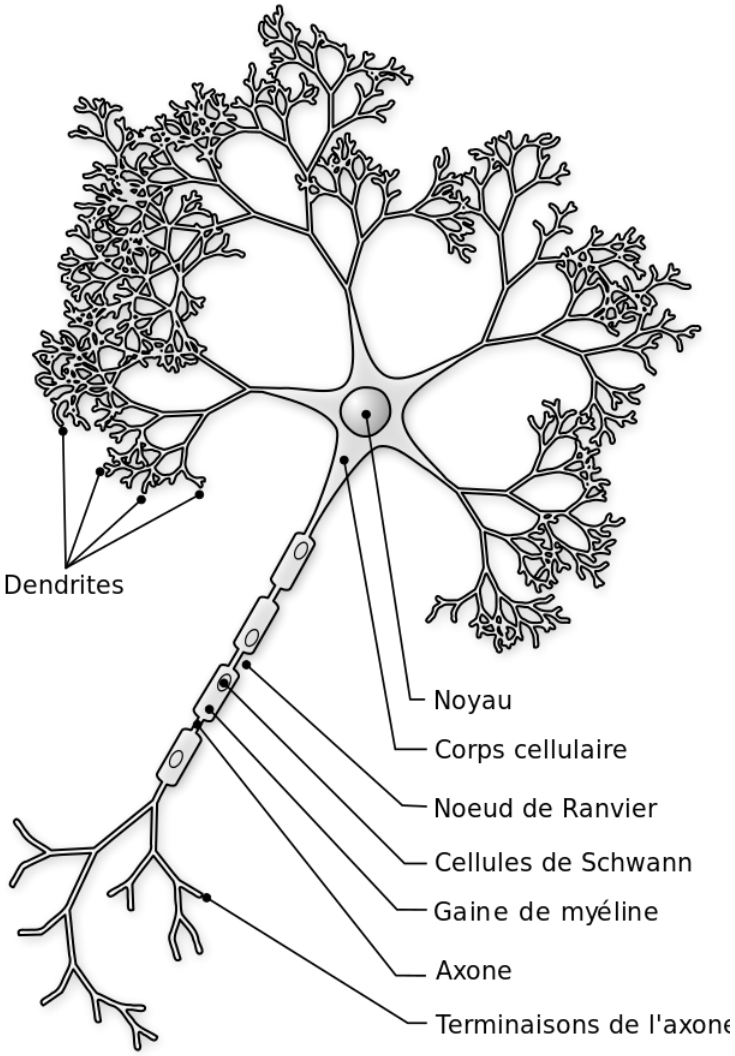
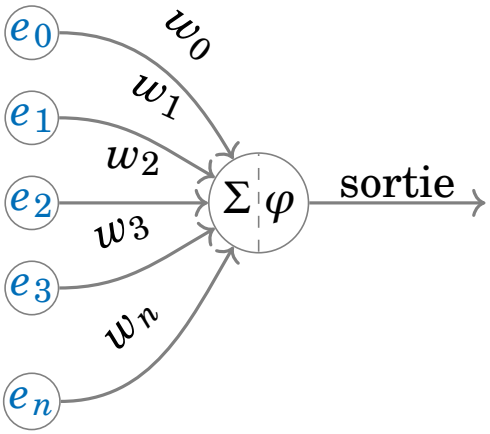


FIGURE II.2 – Le neurone formel



Équation : Neurone formel expression de la sortie

$$sortie = \varphi(z) = \varphi(\sum_{j=1}^{n} w_j.x_j + w_0) \tag{II.2}$$

La fonction d'activation du neurone peut prendre différentes formes suivant l'application.

Sigmoïde : $\varphi(z) = \frac{1}{1+e^{-z}}$

Tangente hyperbolique : $\varphi(z) = \frac{1-e^{-z}}{1+e^{-z}}$

ReLu (Rectified Linear unit) : $\varphi(z) = max(0,z)$

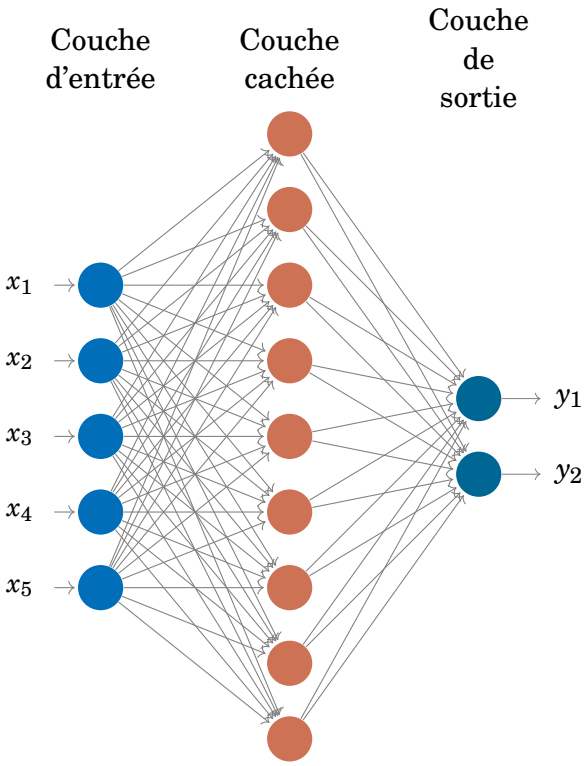
Le physiologiste canadien Donald Hebb fournit une méthode pour modifier les poids synaptiques appelée règle de Hebb [Shaw, 1986]. Il a imaginé un mécanisme qui permet de modifier la valeur des coefficients synaptiques en fonction de l'activité des entités connectées ensembles. Cette méthode de modification des poids du neurone est encore aujourd'hui utilisée couramment.

2 Le perceptron

Le neurone formel est associé à d'autres neurones afin de constituer un réseau de neurones et ainsi augmenter la complexité de résolution. Le perceptron, représenté sur la figure II.3 a été introduit par Frank Rosenblatt en 1957. Il s'agit du premier système automatique capable d'apprendre de ces expériences grâce à une implémentation du concept d'induction. Le perceptron est un réseau de type feedforward, c'est-à-dire que l'information ne se propage que dans un sens, de la couche d'entrée vers la couche de sortie. Malheureusement, le per-

ceptron ne permet pas de traiter les problèmes non linéaires, ce qui constitue une limitation importante.

FIGURE II.3 – Le perceptron



3 Le perceptron multi-couches

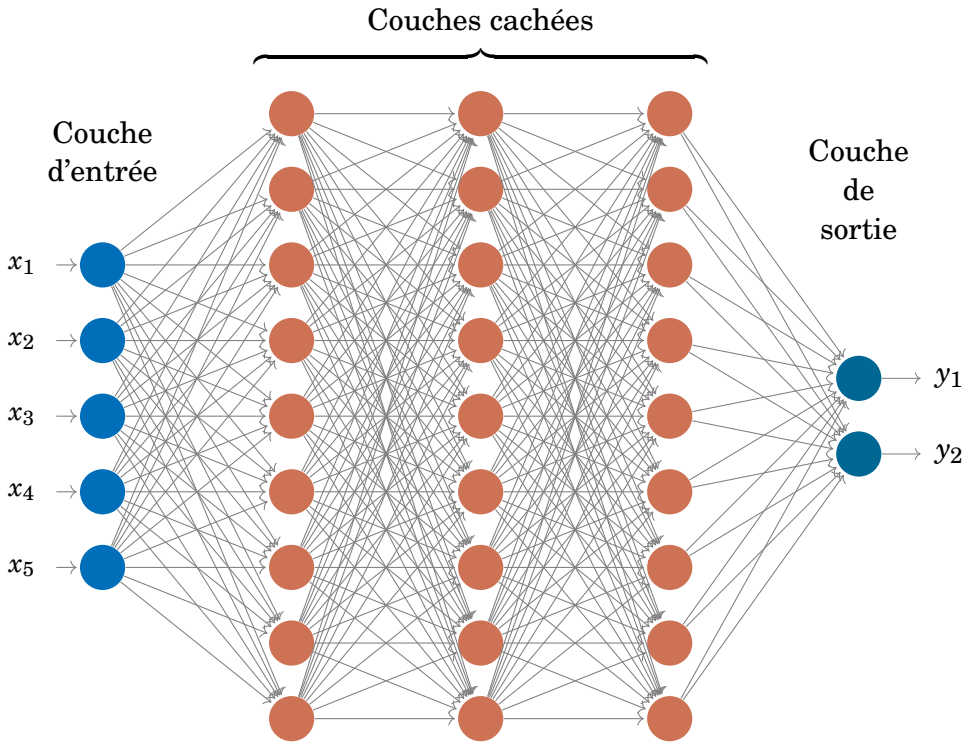
Dans les années 80, l'apparition de la rétro-propagation du gradient de l'erreur permet la création du perceptron multi-couches. Les réseaux de neurones multi-couches ont une topologie en trois parties : une première couche connectée à l'extérieur du réseau, une ou plusieurs couches cachées connectées séquentiellement à partir de la couche d'entrée et ensuite une couche de sortie. On parle de réseau profond lorsque les couches cachées sont nombreuses.

Soit $x = (x_1, x_2, ..., x_n)$ le vecteur d'entrée d'un perceptron à L couches, exprimons h_1 le vecteur de sortie de la première couche.

Équation : Perceptron expression de la sortie de la couche d'entrée

$$h_1 = \varphi(W_{0,1}.x + b_1) \tag{II.3}$$

FIGURE II.4 – Le perceptron multi-couches



avec $W_{0,1}$ la matrice des poids de la couche d'entrée en regard aux vecteurs d'entrée, b_1 le biais et φ la fonction d'activation.

Nous pouvons généraliser cette écriture aux couches suivantes.

Équation : Perceptron expression de la sortie de la couche cachée

$$h_{l+1} = \varphi(W_{l,l+1}.h_l + b_{l+1}) \tag{II.4}$$

La sortie du réseau sera alors calculable comme suit.

Équation : Perceptron expression de la sortie du perceptron

$$y = h_L = \sigma(W_{L-1,L}.h_{L-1} + b_L) \tag{II.5}$$

avec σ la fonction d'activation de la couche de sortie, et $y = (y_1, y_2, ..., y_m)$ le vecteur de sortie du réseau de neurones pour une tache de classification mono-étiquette de m classes $C = (c_1, c_2, ..., c_m)$.

La fonction d'activation de la couche de sortie est différente de celle des couches cachées. En effet, le rôle de chaque couche est différent ainsi que son implémention. La dernière couche pour une tâche de classification permettra la production des probabilités des classes pour les échantillons mis en entrée.

Généralement, pour une classification mono-étiquette, la fonction d'activation de la couche de sortie du réseau est de type Softmax. Cependant, ce n'est pas obligatoire.

Équation : Fonction de coût : Softmax

$$Softmax(z_i) = \frac{e^{zi}}{\sum_{j=1}^{m} e^{zj}} \tag{II.6}$$

avec $z_i = W.h_{i-1} + b$.

y le vecteur de sortie du réseau est alors une variable aléatoire représentant la probabilité du vecteur x d'appartenir à la classe c_i que l'on note $P(y = i|x)$. La fonction Softmax, par sa construction ainsi que la normalisation qu'elle conduit, permet d'obtenir une somme des termes égale à un.

Équation : Les neurones expriment une probabilité

$$\sum_{i \in 1,2,...,m} P(y = i|x) = 1 \tag{II.7}$$

avec m le nombre de neurones de la couche de sortie L.

Pratiquement, la prédiction de la classe pour l'échantillon x est obtenue par la valeur maximale parmi les éléments du vecteur y.

Équation : Estimation de l'étiquette

$$\underset{i \in 1,2,...,m}{argmax}(Softmax(z_L)) \tag{II.8}$$

4 Les réseaux récurrents

Les (RNN) Réseau de Neurones Récurrents sont des réseaux de neurones dans lesquels l'information peut se propager d'avant en arrière, y compris des couches profondes aux couches plus en amont. En cela, ils sont plus proches du vrai fonctionnement du système nerveux des humains, qui n'est pas à sens unique. Ces réseaux ont des connexions récurrentes dans le sens où ils conservent des informations en mémoire. Ils peuvent prendre en compte à un instant t un certain nombre d'états passés. On peut faire une représentation de la succession des états comme sur la figure II.5. Les RNN présentent un caractère dynamique du fait que leurs poids dépendent non seulement des entrées apprises, mais également des sorties précédentes. Pour ces raisons, les RNN sont particulièrement adaptés aux applications impliquant le contexte, la relation d'ordre des données et plus particulièrement au traitement de séquences temporelles telles que l'apprentissage et la génération de signaux, c'est-à-dire lorsque les données forment

une séquence et ne sont pas indépendantes dans leurs successions. D'un point de vue théorique, les RNN ont un potentiel beaucoup plus important que les réseaux neuronaux conventionnels. Ils sont « Turing-complete » , c'est-à-dire qu'ils permettent théoriquement de simuler l'ensemble des algorithmes calculables. Une amélioration importante des réseaux RNN a été apportée avec le concept de bidirectionnalité intégrant une passe « avant » (forward) de la séquence des données dans un sens suivie d'une passe « arrière » (backward) des données de la séquence en sens opposé. Le transfert bidirectionnel d'informations rend leur conception assez compliquée.

Équation : RNN et bidirectionnalité

$$(forward)\, h_t^f = \varphi(W_h^f . h_{t-1}^f + W_x^f . x_t + b_h^f) \tag{II.9}$$

$$(backward)\, h_t^b = \varphi(W_h^b . h_{t-1}^b + W_x^b . x_t + b_h^b) \tag{II.10}$$

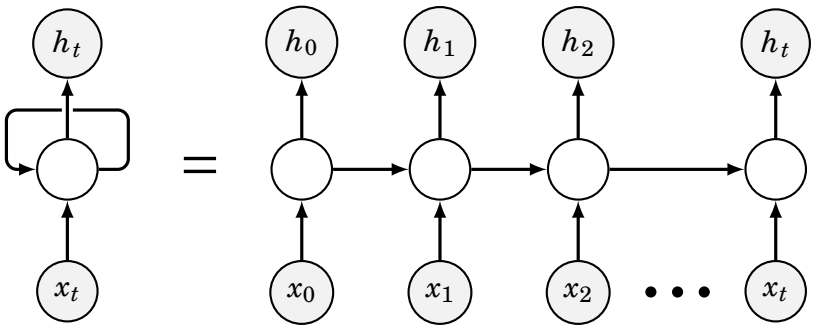
Avec W_h^f le poids avant et W_h^b le poids arrière. Ensuite, la couche de sortie se calcule ainsi.

Équation : Sortie d'un RNN bidirectionnel

$$y(t) = \sigma(W_h^f . h_t^f + W_h^b . h_t^b + b_y) \tag{II.11}$$

σ étant la fonction d'activation de la couche de sortie et b_y les biais.

FIGURE II.5 – Réseau de neurones récurrent déplié

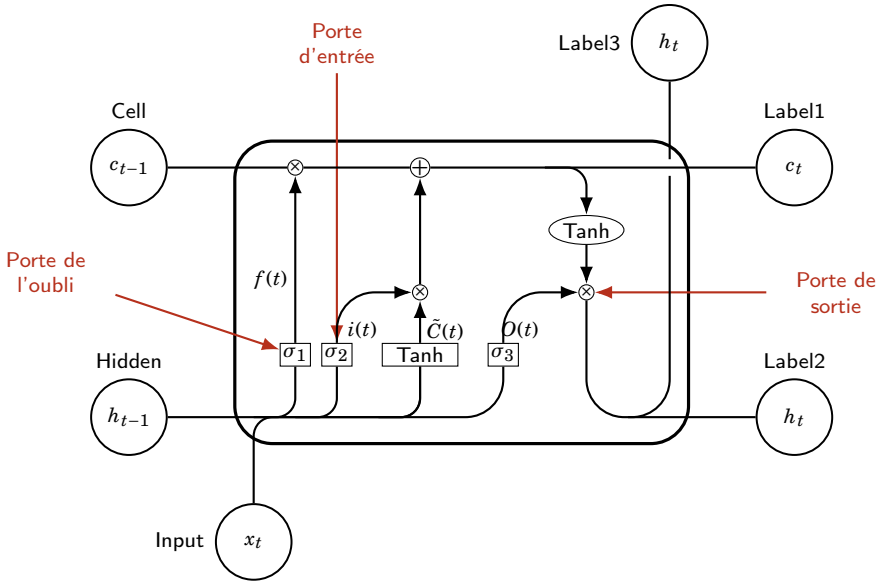


Une des limitations importantes des RNN est leur incapacité à capturer les dépendances à long terme. Ceci est du à la difficulté de la diffusion de la rétroprogation du gradient de l'erreur à travers la boucle de rétroaction.

5 Les réseaux LSTM

Les réseaux RNN sont peu efficaces pour les applications impliquant de longues différences de temps, typiquement la classification des séquences vidéo. Leur « mémoire à court terme » n'est pas suffisante. En effet, les RNN classiques (simples réseaux de neurones récurrents ou RNN vanilla) ne sont pas capables de mémoriser ce que le passé exprime à long terme et ils oublient après une cinquantaine d'itérations. À la fin des années 90, afin de résoudre ces problèmes, des méthodes efficaces ont été développées comme les réseaux LSTM. Ces réseaux à grande « mémoire à court terme » ont notamment révolutionné la reconnaissance de la voix par les machines (reconnaissance vocale) ou la compréhension et la génération de textes.

FIGURE II.6 – Architecture d'une couche LSTM



Un réseau LSTM comporte une cellule de mémoire, typiquement une couche de neurones, ainsi que trois portes : une porte d'entrée, une porte de sortie et une porte de l'oubli. Ces trois portes vont permettre de moduler le flux d'informations à l'entrée, à la sortie et à mémoriser de manière analogique grâce à une fonction d'activation de type sigmoïde. La figure II.6 est une représentation dépliée du fonctionnement d'un réseau LSTM avec ses trois principales composantes.

Un réseau LSTM est basé sur une architecture qui lui permet d'oublier les informations inutiles. La couche sigmoïde 1 prend l'entrée $x(t)$ et $h(t-1)$ et décide quelles parties de l'information de la précédente sortie doivent être supprimées (en sortant un 0). Cette porte est appelée « oubli » . La sortie de cette porte est $f(t) * C(t-1)$.

Équation : LSTM porte de l'oublie

$$f(t) = \sigma_1(W_f.[h[t-1, x_t] + b_f]) = \begin{cases} 0 \rightarrow & se\ débarrasser\ du\ contenu\ c_t \\ 1 \rightarrow & conserver\ le\ contenu\ de\ c_t \end{cases} \quad \text{(II.12)}$$

L'étape suivante permet de décider puis de stocker les informations de la nouvelle entrée $x(t)$ dans l'état de cellule. Une couche sigmoïde décide laquelle des nouvelles informations doit être mise à jour ou ignorée. Une couche tanh crée un vecteur de toutes les valeurs possibles à partir de la nouvelle entrée. Ces deux sont multipliées pour mettre à jour le nouvel état de cellule. Cette nouvelle mémoire est ensuite ajoutée à l'ancienne mémoire $C(t-1)$ pour donner $C(t)$.

Équation : LSTM cellule de mémoire

$$i(t) = \sigma_2(W_i.[h[t-1, x_t] + b_i]) = \begin{cases} 0 \rightarrow & pas\ de\ mise\ à\ jour \\ 1 \rightarrow & mise\ à\ jour \end{cases} \quad \text{(II.13)}$$

$$\tilde{C}(t) = tanh(W_c.[h[t-1, x_t] + b_c]) \quad \text{(II.14)}$$

$$C(t) = C(t-1) + i(t) * \tilde{C}(t) \quad \text{(II.15)}$$

Enfin, il faut décider ce qui va être produit. Une couche sigmoïde décide quelles parties de l'état de la cellule vont être générées. Ensuite, l'état de la cellule à travers un tanh est fixé générant toutes les valeurs possibles et il est multiplié par la sortie de la porte sigmoïde, de sorte que la sortie produise uniquement les parties décidées. Notre modèle n'apprend pas la réponse de la dépendance immédiate, mais plutôt de la dépendance à long terme.

Équation : LSTM porte de sortie

$$O(t) = \sigma_3(W_o.[h[t-1, x_t] + b_o]) = \begin{cases} 0 \rightarrow & pas\ de\ sortie \\ 1 \rightarrow & retourne\ l'intégralité\ de\ la\ cellule \end{cases} \quad \text{(II.16)}$$

Nous venons de voir qu'il y a une grande différence entre l'architecture d'un RNN et d'un LSTM. Dans un LSTM, le modèle apprend quelles informations stocker dans la mémoire à long terme et lesquelles oublier. L'entraînement des réseaux LSTM est relativement rapide, quelques « époques » [3], et la remontée du gradient de l'erreur est fonctionnelle, cependant la mise en parallèle des calculs n'est pas possible du fait de la prise en compte de la relation d'ordre de l'informa-

3. Une époque correspond à une passe complète de l'ensemble des vecteurs d'entrée.

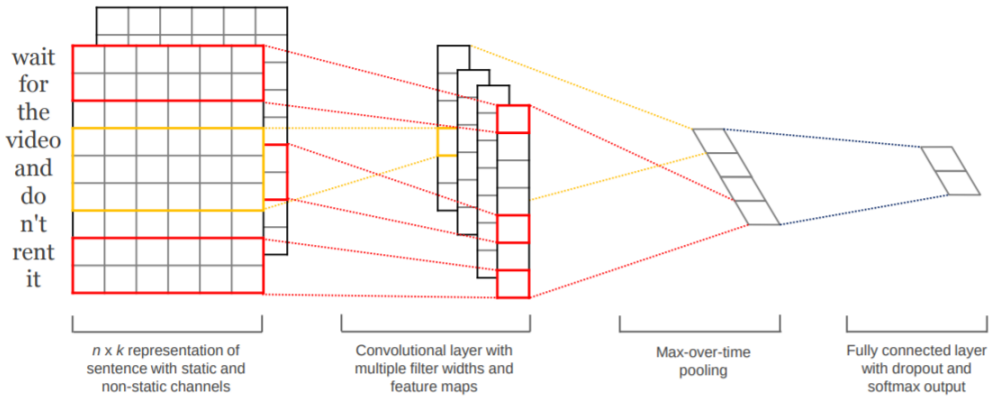
tion. Ces approches ont beaucoup été utilisées pour le traitement automatique du langage [Ghosh et al., 2016], l'analyse d'images [Stollenga et al., 2015] ou encore le diagnostique médical [Kobold, 2019].

6 Les réseaux de convolution

Les réseaux de neurones CNN sont construits pour une analyse fine des données exprimées sous la forme d'une grille, typiquement les pixels d'une image. Ils ont été proposés par [Krizhevsky et al., 2012] et ont montré leur efficacité sur l'analyse de textes ou le traitement d'images.

Un réseau CNN est construit par la succession d'une couche de convolution et d'une couche d'aggrégation de l'information (Pooling) et se termine par une couche de neurones totalement connectés. Sur la figure II.7, on peut retrouver l'architecture d'un réseau CNN pour une tâche de classification de textes.

FIGURE II.7 – Représentation de la structure d'un réseau CNN pour une analyse de phrases [Kim, 2014]



Une convolution de matrice revient à multiplier deux matrices de dimension incompatible. L'opération de convolution consiste à appliquer un filtre ou « kernel » sur la grille ou matrice de données. Ce filtre est une matrice de dimension inférieure à la matrice de données. On positionne le centre du noyau sur l'élément de la matrice que l'on souhaite modifier et on applique un simple produit scalaire entre le recouvrement par le kernel sur la matrice et le kernel lui-même. Puis, on déplace le noyau d'un pas sur l'élément suivant et ainsi de suite. On peut retrouver une représentation de cette heuristique dans la figure II.8. Cela permet de faire émerger pour chaque filtre choisi, une analyse de caractéristique particulière. Il est courant d'appliquer un grand nombre de filtres sur un réseau de convolution ce qui produit une carte de caractéristiques. Il est

évident que cette opération génère une explosion de la quantité de valeurs obtenues. On associe alors un sous-échantillonnage de manière à réduire la quantité d'informations.

Le Pooling est une méthode permettant de prendre une large grille d'informations et d'en réduire la taille tout en préservant les informations les plus importantes. L'algorithme consiste, à faire glisser une petite fenêtre pas à pas sur toutes les parties de la grille et à prendre la valeur maximum, la moyenne ou le minimum de cette fenêtre à chaque pas. La grille obtenue est par conséquent de dimension réduite par rapport à la grille initiale. Le résultat de cette opération permet au réseau de trouver si une information judicieuse est présente dans la grille, sans se soucier de l'endroit précis où cette information se trouve.

Le paramétrage d'un réseau CNN n'est pas une tâche aisée. Ainsi, il faut décider des multiples caractéristiques à prendre en compte de manière empirique : Combien de filtres faut-il utiliser ? Quelles dimensions pour ces filtres ? Pour chaque couche de Pooling, quelle taille de fenêtre doit-on choisir ? Quel pas ? Pour chaque couche entièrement connectée supplémentaire, combien de neurones cachés doit on définir ?

Cependant, un avantage important des réseaux CNN est la facilité de mise en parallèle des calculs. Aussi, ces réseaux ont été fortement utilisés dans les tâches de traitement automatique du langage en apprentissage profond [Zhang et al., 2016b; Abulaish and Sah, 2019].

7 Les auto-encodeurs

Les auto-encodeurs cherchent par leur structure à être capables de reconstruire à l'identique une entrée vectorielle. L'intérêt de ces réseaux est de tirer partie de l'information apprise pour résoudre cette tâche en apparence triviale. Une fois l'auto-encodeur entraîné, il pourra être utilisé pour diverses tâches comme la réduction de dimension [Anam and Al-Jumaily, 2015; Wang et al., 2016] ou encore la traduction de texte [Zhang et al., 2016a]. Les auto-encodeurs pour la phase d'entraînement utilisent en sortie la même information qu'en entrée. En cela, on parle d'apprentissage non supervisé car ils ne demandent aucune autre information additionnelle que les vecteurs d'entrée. Comme on peut le voir sur la figure II.9 la couche d'entrée possède le même nombre de neurones que la couche de sortie. L'architecture neuronale se situant entre ces deux couches, peut être de différents types : multiples couches totalement connectées, couche CNN, couche LSTM, etc. Sur la figure II.9, c'est une simple couche de neurones.

La structure d'entrée sert à projeter les vecteurs d'entrée dans l'espace intermédiaire latent.

FIGURE II.8 – Représentation du calcul de convolution d'une grille 5 par 5 avec un noyau en 3 par 3 et un pas de 1 [Yamashita et al., 2018]

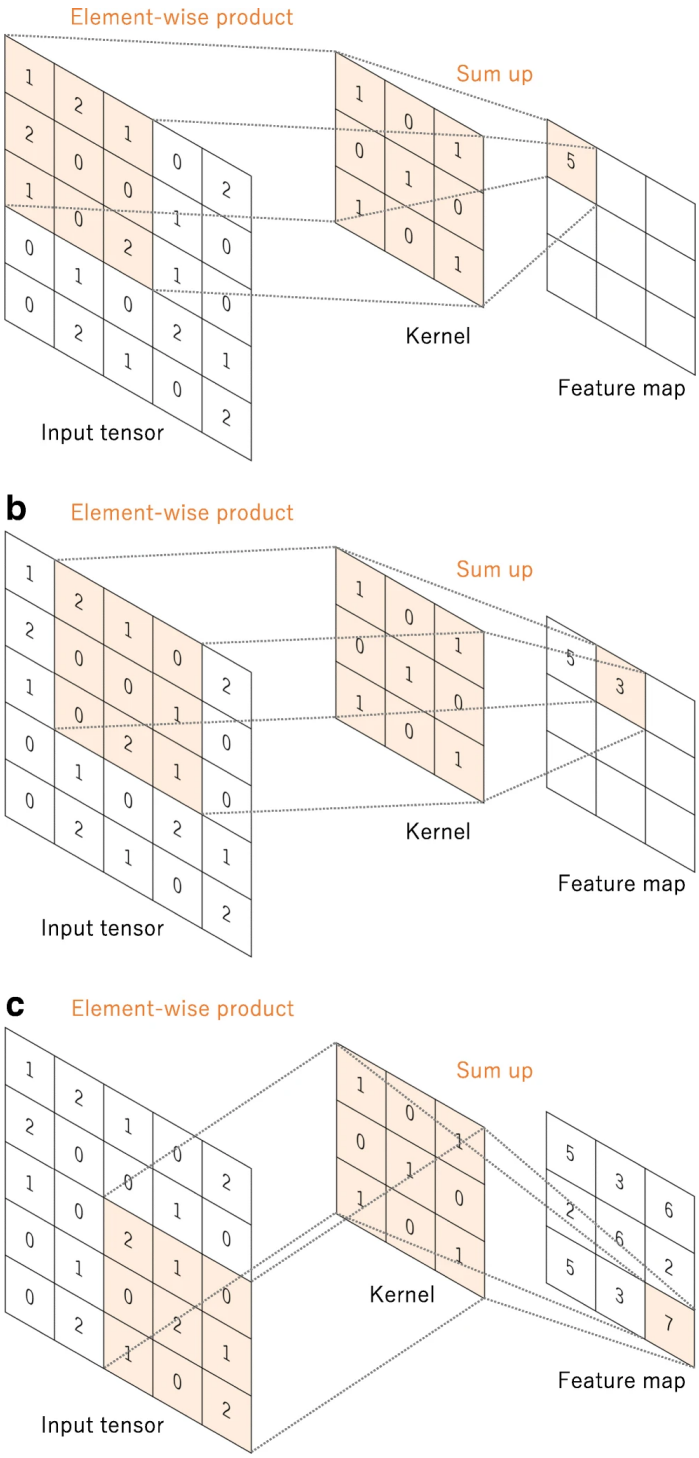
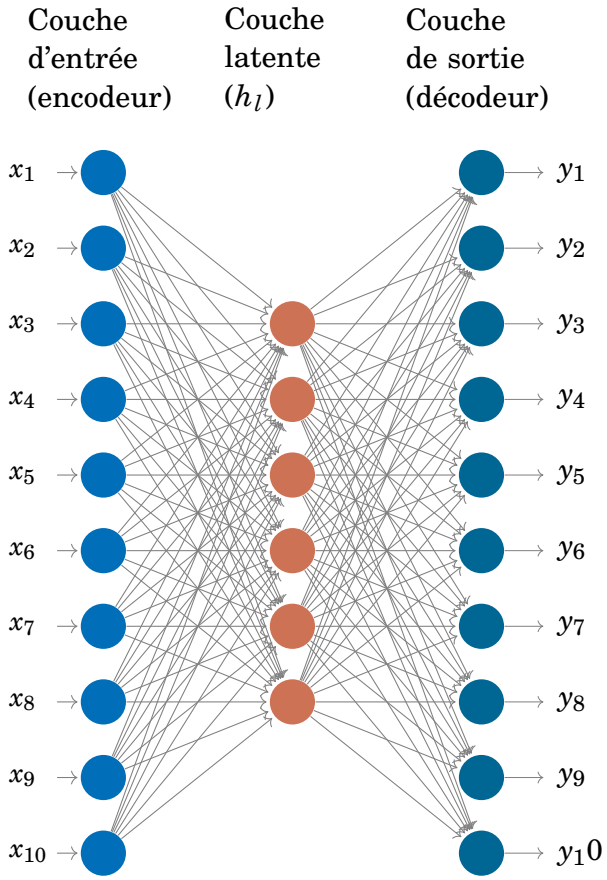


FIGURE II.9 – Architecture d'un auto-encodeur



Équation : Auto-encodeur : espace latent

$$h_l = \varphi_{enc}(W_{enc}.X + b_{enc}) \tag{II.17}$$

Avec φ_{enc} la fonction d'activation de l'encodeur, W_{enc} la matrice de poids de l'encodeur et b_{enc} vecteur de biais.

Le décodeur est ensuite utilisé pour reconstruire le vecteur de sortie le plus identiquement possible à celui fourni en entrée.

Équation : Auto-encodeur : sortie du décodeur

$$y = \varphi_{dec}(W_{dec}.h_l + b_{dec}) \tag{II.18}$$

Avec φ_{dec} la fonction d'activation du décodeur, W_{dec} la matrice de poids du décodeur et b_{dec} le vecteur de biais.

Généralement, l'erreur moyenne quadratique est utilisée comme fonction de coût.

Équation : Auto-encodeur : fonction de coût

$$E(X,Y) = \frac{1}{N}\sum_{i=1}^{N} ||X_i - Y_i||^2 \tag{II.19}$$

Où N représente le nombre d'échantillons.

8 Le mécanisme d'attention

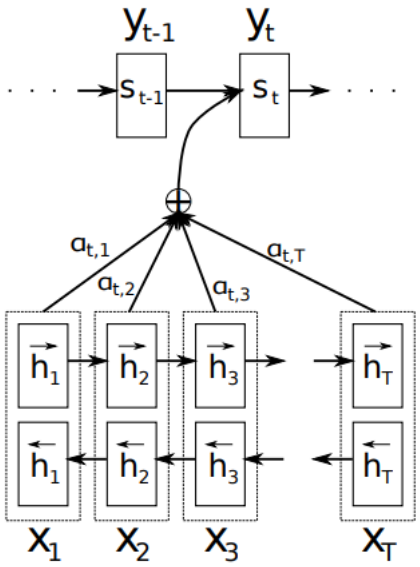
En psychologie, l'attention est le processus cognitif de concentration sélectif sur un ou plusieurs objets tout en ignorant les autres.

Le mécanisme d'attention est apparu en vue de faire progresser les systèmes de traduction automatique basés sur des auto-encodeurs dans le TALN [Bahdanau et al., 2015]. Plus tard, ce mécanisme, ou ses variantes, ont été utilisés dans d'autres applications, notamment la vision par ordinateur [Parmar et al., 2019], le traitement de la parole [Simonnet, 2019], etc.

Dans les applications de traduction automatique [Do et al., 2015], il est courant d'utiliser des auto-encodeurs sur la base d'une architecture LSTM. Chaque fois que le modèle proposé génère une phrase, il recherche un ensemble de positions dans les couches cachées de l'encodeur où les informations les plus pertinentes sont disponibles. Cette idée est appelée « Attention » .

Le LSTM bidirectionnel utilisé dans la figure II.10 génère une séquence d'annotations $(h_1, h_2, ..., h_{Tx})$ pour chaque phrase d'entrée. Tous les vecteurs $h_1, h_2, ...$, etc., utilisés dans le processus sont la concaténation des états cachés de sortie des LSTM avant et arrière dans l'encodeur.

FIGURE II.10 – Le mécanisme d'attention [Bahdanau et al., 2015]



Équation : Attention encodeur

$$h_j = [\overrightarrow{h_j^T}, \overleftarrow{h_j^T}] \tag{II.20}$$

Tous les vecteurs $h_1, h_2, h_3, ..., h_{Tx}$ sont des représentations du nombre Tx de mots dans la phrase d'entrée. Dans le modèle de l'encodeur et du décodeur simple, seul le dernier état du LSTM de l'encodeur a été utilisé (h_{Tx} dans ce cas) comme vecteur de contexte.

Le vecteur de contexte c_i pour le mot de sortie y_i est généré en utilisant la somme pondérée des annotations.

Équation : Attention vecteur de contexte

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij}.h_j \tag{II.21}$$

Les poids α_{ij} sont calculés par une fonction Softmax.

Équation : Attention Softmax

$$\alpha_{ij} = \frac{exp(e_{ij})}{\sum_{k=1}^{T_x} e_{ik}} \; e_{ij} = a(S_{i-1}, h_j) \tag{II.22}$$

e_{ij} est le score de sortie d'un réseau neuronal décrit par la fonction a qui tente de capturer l'alignement entre l'entrée en j et la sortie en i.

Il en résulte que α est un vecteur. Ses éléments sont les poids correspondant à chaque mot dans la phrase d'entrée en vis à vis de leur importance dans le contexte.

L'attention permet de regarder la totalité d'une phrase, pour établir des liens entre un mot particulier et son contexte. Ceci est très différent des RNN à courte mémoire, focalisés en amont, et également très différent des réseaux convolutifs focalisés sur la proximité de l'information analysée.

Actuellement, le mécanisme d'attention est très largement utilisé dans un certain nombre d'applications comme : les soins, la santé, la reconnaissance vocale [Etienne, 2019], les systèmes de recommandation ainsi que les voitures autonomes [Chen et al., 2019].

9 L'architecture de réseau neurone Transformer

L'architecture de réseau neuronal Transformer proposée par Vaswani et al. [2017a] a marqué l'une des percées majeures ces dernières années dans le domaine de l'analyse de textes en apprentissage profond. Les couches d'attention de l'architecture Transformer alignent les mots d'une séquence avec d'autres mots de la séquence, proposant ainsi une représentation de la séquence. Il est non seulement plus efficace en termes de représentation, mais aussi plus efficace en termes de prédiction par rapport aux architectures de la littérature de type RNN et CNN.

L'attention multi-têtes utilisée ici est un processus essentiellement constitué de plusieurs couches d'attention apprenant conjointement différentes représentations à partir de différentes positions.

Équation : Transformer Attention

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{II.23}$$

10 Conclusion

Nous venons dans ce chapitre de faire la description des principales architectures utilisées en apprentissage profond dans le traitement automatique du langage, à savoir les réseaux de convolution, les réseaux LSTM, les auto-encodeurs ainsi que l'architecture Transformer. Toutes ces architectures ont des comportements différents, aussi il est important de savoir si une architecture se détache pour le traitement des données que nous cherchons à classifier. Quel classifieur

FIGURE II.11 – (à gauche) produits scalaires de l'attention. (à droite) Multi-Head Attention se composant de plusieurs couches d'attention fonctionnant en parallèle [Vaswani et al., 2017b]

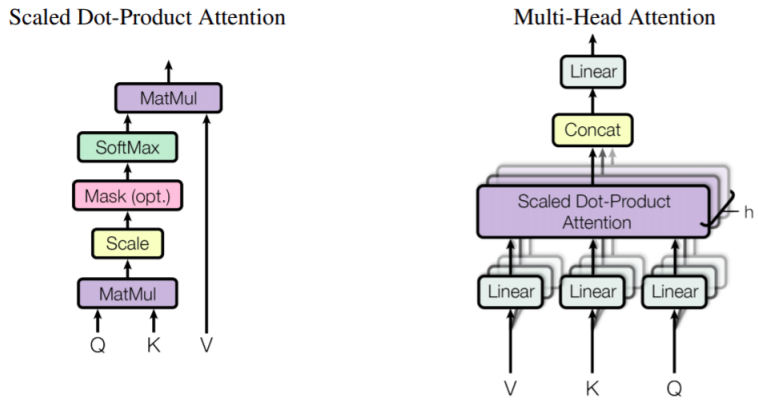
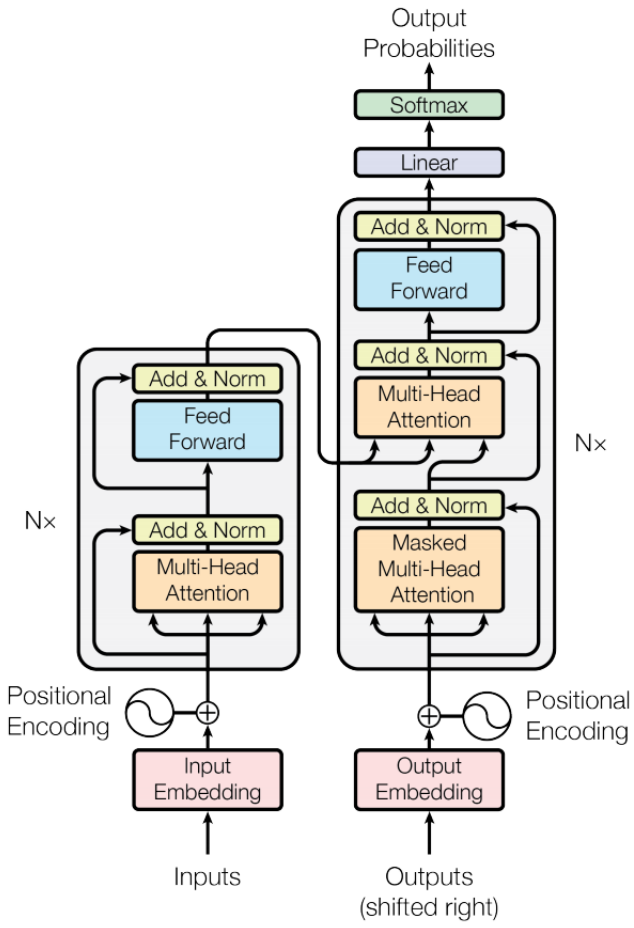


FIGURE II.12 – Le mécanisme Transformer [Vaswani et al., 2017b]



est le plus adapté aux petites bases de données textuelles annotées ? Quelle mise en œuvre est elle la moins complexe et la plus efficace ? Quelle préparation des données est-elle nécessaire ? L'évaluation du processus de classification a t-il un impact ? Nous poursuivons notre manuscrit par une étude expérimentale et comparative de ces algorithmes, en classification de texte, dans le but d'obtenir des éléments de réponses à ces interrogations.

III

Comparaison des classifieurs de textes

Plan du chapitre

La classification automatique de textes est un sujet classique de Traitement Automatique de la Langue (TAL), qui consiste à assigner des catégories prédéfinies à des documents en texte libre. Pour une tâche précise, on cherche à identifier le meilleur algorithme de classification et à définir les meilleures caractéristiques prises en entrée de ces classifieurs. Ces modèles ont été appliqués avec succès sur des données avec de très nombreuses caractéristiques dimensionnelles parfois éparses.

Dernièrement, les méthodes d'apprentissage profond se sont révélées très efficaces et cette tendance s'est confirmée avec le succès des représentations continue de mots (word embeddings) [Mikolov et al., 2010, 2013a]. Collobert et al. [2011] ont démontré qu'une architecture d'apprentissage profond, même simple, surpasse la plupart des approches classiques pour des tâches variées de TAL telles que la reconnaissance d'entités nommées (NER) [Nguyen et al., 2016], le découpage [Vinyals et al., 2015; Zhu et al., 2013], l'étiquetage de rôles sémantiques (SRL) (Semantic Role Labeling) [He et al., 2017; Zagoruyko and Komodakis, 2017], le marquage POS [Andor et al., 2016; Kumar et al., 2016], la classification de sentiments [Rosenthal et al., 2019; Kalchbrenner et al., 2014; Kim, 2014], la traduction automatique [Sukhbaatar et al., 2015]. D'autres articles montrent l'apport de ces méthodes pour des tâches de classification de textes [Le et al., 2017], de clustering de textes [Xu et al., 2017], ou encore de reconnaissance de formes (pattern recognition) [Baccouche et al., 2011].

Dans ce chapitre, nous allons comparer l'efficacité des méthodes de classification traditionnelles avec plusieurs architectures d'apprentissage profond.

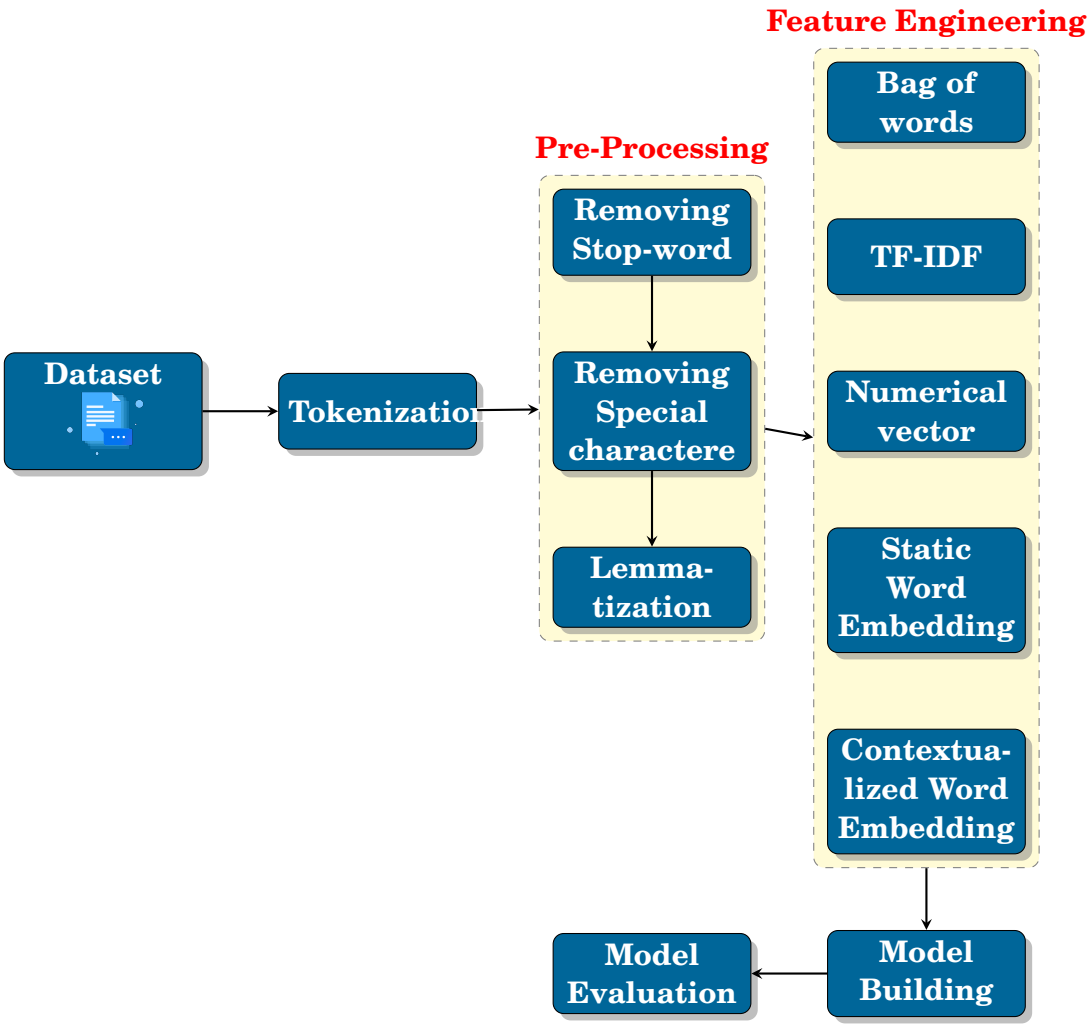
Le chapitre est organisé en trois sections. Dans la section 1, nous présentons un état de l'art des algorithmes de classification automatique de textes. Dans la section 2, nous présentons un descriptif de la méthode expérimentale de notre étude. Dans la section 3, nous présentons les méthodes d'évaluation utilisées dans nos expérimentations. Dans la section 4, nous discutons ces résultats et ouvrons des perspectives.

1 État de l'art

La classification de textes est l'une des tâches importantes en traitement automatique de la langue. Il s'agit d'une approche supervisée, pour laquelle nous disposons de données étiquetées pour apprendre un modèle. Elle consiste à identifier la catégorie (ou classe) d'un texte donné à partir d'une liste finie de catégories. Elle peut s'appliquer à tous types de textes tels que les blogs, les livres, les pages Web, les articles d'actualité ou les tweets. Il existe diverses applications telles que la détection des spams [Crawford et al., 2016], la catégorisation des

tâches dans les services CRM (Customer Relationship Management) [Roussinov and Zhao, 2004], la catégorisation des produits sur les sites Web des détaillants en ligne [Zhang and Paramita, 2019], la classification du contenu des sites en ligne [Fidalgo et al., 2019], les sentiments des commentaires [Krawczyk et al., 2017], etc. Dans la section suivante, nous présentons l'état de l'art du processus complet de classification de textes.

FIGURE III.1 – Processus d'analyse automatique de textes



1.1 Représentation des documents

La figure III.1 représente le processus d'analyse automatique des textes. Ces derniers doivent préalablement être pré-traités. Selon la tâche, on peut supprimer les termes vides, enlever les caractères spéciaux, lemmatiser, etc. Ensuite, il est nécessaire de représenter ces textes, qui sont une chaîne de caractères,

par des caractéristiques qui seront prises en entrée des algorithmes de classification. Cette représentation des textes a un fort impact sur la précision et la généralisation du système d'apprentissage. Dans la suite de cette section, nous allons lister les différentes représentations des textes existantes.

1.1.1 Représentation par sac de mots

La méthode de représentation la plus simple est la représentation par sac de mots. Considérons un dictionnaire $W = \{w_1, ...w_n\}$ composé de tout ou partie des mots w_i du corpus. Un document doc est représenté par un vecteur $d = \{d_1, ...d_n\}$ correspondant aux nombres d'occurrences des mots qu'il contient. Les mots présents dans le dictionnaire et absents dans le document auront pour valeur 0. Considérons l'exemple ci-dessous :

Exemple 1.1 [Documents]

document 1 Pourquoi un geek se suicide ? Pour mettre sa vie en mode sans échec.

document 2 Où partent les geeks en vacances ?- Aux C-Shell.

document 3 Un geek ne crie pas, un geek URL.

Ici chaque mot d'un document est représenté par un seul nombre entier dans un document. Nous pouvons voir sur la table III.1 la présence d'un grand nombre de zéros.

TABLE III.1 – Document-Term Matrix (DTM) : sac de mots.

Dictionnaire	Document 1	Document 2	Document 3
crier	0	0	1
geek	1	1	2
mettre	1	0	0
mode	1	0	0
partir	0	1	0
pourquoi	1	0	0
sans	1	0	0
shell	0	1	0
suicide	1	0	0
url	0	0	1
vacances	0	1	0
vie	1	0	0
échec	1	0	0

La représentation en sacs de mots d'un corpus de documents implique que la matrice de représentation de l'ensemble des documents a pour dimension le nombre de mots distincts dans le corpus. Ce nombre est généralement très important. La plupart des valeurs de la matrice sont des zéros surtout si les textes sont courts. Pour cette raison, les sacs de mots sont généralement des ensembles de données clairsemés (ou sparse) de grande dimension. Cela peut poser des problèmes, notamment pour faire une comparaison efficace des vecteurs de description des documents.

1.1.2 Représentation par TF-IDF

Comme pour la représentation précédente, chaque document doc est associé à un vecteur $d = (d_1, ..., d_n)$ de sorte que les documents similaires auront une représentation similaire sur un même corpus (selon une métrique de similitude fixe). Chaque élément d_i représente une information sur un mot distinct w_i. Pour un document doc le vecteur d est calculé dans l'équation III.1 :

Équation : TF-IDF

$$d = TF(w_i, doc) \times IDF(w_i) \tag{III.1}$$

Le terme fréquence $TF(w_i, doc)$ est le nombre de fois où le mot w_i apparaît dans le document doc.

La fréquence inverse du document $IDF(w_i)$ peut être calculée à partir de la fréquence des mots dans le corpus selon l'équation III.2.

Équation : IDF

$$IDF(w_i) = log(\frac{|D|}{DF(w_i)}) \tag{III.2}$$

Le terme $DF(w_i)$ est le nombre de documents dans lesquels le mot w_i apparaît au moins une fois. $|D|$ est le nombre total de documents.

Intuitivement, la fréquence inverse du document d'un mot est faible si elle apparaît dans de nombreux documents et est plus élevée si le mot n'apparaît que dans un seul.

Finalement, la métrique de pondération des mots d_i exprime le fait qu'un mot w_i est un terme important pour le document s'il se produit fréquemment à l'intérieur de celui-ci. D'autre part, les mots qui apparaissent dans de nombreux documents du corpus sont des termes d'indexation moins importants en raison de leur faible fréquence inverse du document [Salton and Buckley, 1988].

Reprenons l'exemple précédent. Comme précédemment, nous pouvons voir sur le tableau III.2 la présence d'un grand nombre de zéros dans la matrice

Terme-Document (ou Document Term-Matrix (DTM). Si les mots sont bien différenciés par leurs valeurs dans des documents différents du corpus, leur représentation numérique est identique pour un document particulier quelles que soit leurs positions dans celui-ci.

TABLE III.2 – Document-Term Matrix (DTM) : TF-IDF

Dictionnaire	Document 1	Document 2	Document 3
crier	0	0	0.54
geek	0.21	0.32	0.64
mettre	0.36	0	0
mode	0.36	0	0
partir	0	0.54	0
pourquoi	0.36	0	0
sans	0.36	0	0
shell	0	0.54	0
suicide	0.36	0	0
url	0	0	0.54
vacances	0	0.54	0
vie	0.36	0	0
échec	0.36	0	0

1.1.3 Représentation des documents par identifiant

Dans cette représentation, on considère toujours un dictionnaire $W = \{w_1, ...w_n\}$, celui-ci peut être composé d'entités pouvant être des caractères, des parties de mots ou des mots entiers. Le vecteur représentant un document $d = \{d_1, ...d_n\}$ est alors construit par la séquence des indexes des entités qu'il contient dans l'ordre où elles apparaissent dans le document.

Les documents du corpus ayant généralement des tailles variables, en conséquence les représentations vectorielles de ces documents sont également de tailles variables [Lai et al., 2015].

Reprenons l'exemple précédent. On voit dans le tableau III.3 que le document 1 commence par l'entité 6 (pourquoi) suivi de l'entité 2 (geek).... Un mot est alors représenté par un nombre entier ou une succession de nombres entiers. Nous pouvons voir que si la relation d'ordre des mots est bien respectée dans les vecteurs représentant les documents, leur représentation est identique dans l'ensemble du corpus.

TABLE III.3 – Séquence vectorielle

Dictionnaire	index	Document 1	Document 2	Document 3
crier	1	6	5	2
geek	2	2	2	1
mettre	3	9	11	2
mode	4	3	8	10
partir	5	12		
pourquoi	6	4		
sans	7	7		
shell	8	13		
suicide	9			
url	10			
vacances	11			
vie	12			
échec	13			

1.1.4 Représentation statique continue des mots

Les techniques de représentions précédentes souffrent de problèmes majeurs comme la sparcité des matrices ou la malédiction de la dimension. De plus les relations syntaxiques et sémantiques ne sont pas prises en compte. Afin de progresser dans la description des mots et des documents, des techniques de représentation des mots par des vecteurs denses de plus faibles dimensions ont été introduites [Bengio et al., 2003; Schwenk, 2007]. Ces représentations sont appelées word embeddings, neural embeddings ou prediction-based embeddings, ou en français représentation vectorielle du mot, représentation continue de mots mais aussi plongement de mots. Dans ce manuscrit nous choisissons d'utiliser l'expression « **représentation continue des mots** » .

Les premières solutions comme Word2vec Mikolov et al. [2013a] ou encore GloVe Pennington et al. [2014] ont permis une amélioration substantielle des classifieurs neuronaux. Récemment, de nouvelles architectures comme ELMo (Embeddings from Language Models) Peters et al. [2018] ou BERT (Bidirectional Encoder Representations from Transformers) Devlin et al. [2019] ont permis de progresser encore dans cette voie. Nous présentons ces solutions dans les paragraphes suivants.

Dans ce type de représentations, deux mots apparaissant dans un contexte sémantique proche, sont représentés par deux vecteurs numériques proches, comparables par des mesures de similarité. Chaque mot w_i d'un document d sera représenté par un vecteur de nombres réels $w_i = \{w_{i,1}, ...w_{i,m}\}$ de dimension généralement très importante. Comme dans beaucoup de domaines liés à

l'apprentissage profond et aux réseaux de neurones, la puissance de calcul des nouvelles machines a beaucoup amélioré ces représentations ces dernières années.

Il existe deux approches principales pour appréhender cette représentation contextuelle. Une première approche dénombre la fréquence de co-occurrence des mots comme GloVe [Pennington et al., 2014]. Une deuxième approche prédit un mot à partir de son contexte ou inversement comme Word2vec [Mikolov et al., 2013a]. Dans la suite, nous allons détailler ces deux types de représentations.

1.1.4.a GloVe

GloVe Pennington et al. [2014] est un algorithme d'apprentissage non supervisé permettant d'obtenir des représentations vectorielles des mots. La représentation est effectuée sur des statistiques de co-occurrences mot-mot globales à partir du corpus, et les représentations résultantes présentent des sous-structures linéaires intéressantes de l'espace vectoriel de mots.

GloVe est un modèle log-bilinéaire pondéré par les moindres carrés. L'intuition principale sous-jacente au modèle est la simple observation que les ratios de probabilités de co-occurrences mot-mot sont porteuses d'une certaine forme de signification. C'est pour cette raison que les vecteurs de mots résultants fonctionnent bien sur des tâches de comparaison de mots.

L'algorithme GloVe comprend les étapes suivantes :

1. Calculer la matrice de co-occurrences de mots $MCOM$. Un élément de cette matrice $MCOM_{i,j}$ représente la fréquence à laquelle le mot w_i apparaît dans le contexte du mot w_j. Pour calculer cette fréquence, pour chaque terme, nous recherchons des termes dans une zone définie par une fenêtre avant et après le terme. Moins d'importance est accordée aux mots les plus éloignés, en utilisant généralement la formule III.3 [Pennington et al., 2014]. L'éloignement représente la distance entre deux mots d'un document exprimée en nombre de mots.

   Équation : GloVe éloignement

   $$poids = \frac{1}{éloignement} \tag{III.3}$$

2. Définir une contrainte souple pour chaque paire de mots.

   Équation : GloVe contrainte souple

   $$w_i^T \times w_j + b_i + b_j = log(MCOM_{ij}) \tag{III.4}$$

Où b_i et b_j sont les vecteurs biais associés respectivement aux vecteurs w_i et w_j.

3. Définir une fonction de coût

Équation : GloVe fonction de coût

$$J = \sum_{i=1}^{n} \sum_{j=1}^{n} f(MCOM_{ij})(w_i^T w_j + b_i + b_j - log(MCOM_{ij})^2 \quad \text{(III.5)}$$

Où n est la taille du vocabulaire en nombre de mots et T la transposée du vecteur w_i.

Ici f est une fonction de pondération qui aide à limiter l'apprentissage uniquement à partir de paires de mots extrêmement courants. Les auteurs de GloVe utilisent la fonction suivante :

Équation : GloVe fonction de pondération

$$f(MCOM_{ij}) = \begin{cases} (\frac{MCOMij}{x_{max}})^{\alpha} & si \quad MCOM_{ij} < x_{max} \\ 1 & sinon \end{cases} \quad \text{(III.6)}$$

x_{max} est la valeur maximale de co-occurence admise sinon la fonction renvoie directement la valeur 1. Classiquement la valeur de x_{max} est de l'ordre de 100. Dans les autres cas, la valeur renvoyée est comprise entre 0 et 1 et α est un coefficient pour régler la distribution des poids ; une valeur courante est $0,75$.

Dans la table III.4, nous pouvons voir la représentation par GloVe des phrases de l'exemple 1.1. Pour construire cette représentation, nous utilisons les vecteurs de mots pré-entrainés du fichier glove.6B.50d.txt téléchargeable à l'url [1]. On note ici l'absence de représentation pour les mots « mettre » , « échec » , « url » . Cette absence sera préjudiciable à la qualité de représentation des documents.

TABLE III.4 – Représentation continue des mots statiques : GloVe

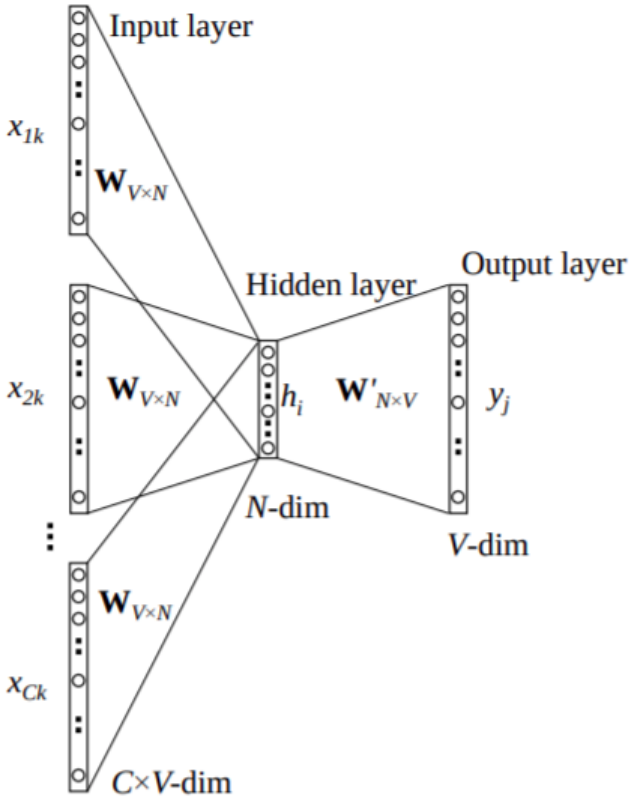
Dictionnaire	index	Document 1	D 1	D 2	D 3	Document 2	D 1	D 2	D 3	Document 3	D 1	D 2	D 3
crier	1	6	-0.306	-1.01	-0.175	5	-0.597	0.477	-1.35	2	-0.912	0.0709	-0.176
geek	2	2	-0.912	0.0709	-0.176	2	-0.912	0.0709	-0.176	1	0.367	0.774	0.997
mettre	3	9	1.17	-0.320	0.606	11	0.340	-0.145	-0.734	2	-0.912	0.0709	-0.176
mode	4	3	_	_	_	8	0.770	-0.598	0.405	10	_	_	_
partir	5	12	0.130	0.495	-0.0366	0	0	0	0	0	0	0	0
pourquoi	6	4	0.166	-0.259	0.0814	0	0	0	0	0	0	0	0
sans	7	7	0.734	-0.181	-1.38	0	0	0	0	0	0	0	0
shell	8	13	_	_	_	0	0	0	0	0	0	0	0
suicide	9												
url	10												
vacances	11												
vie	12												
échec	13												

1. https://nlp.stanford.edu/projects/glove/

1.1.4.b Word2vec

Ces dernières années les méthodes à base de neurones artificiels se sont imposées. Word2vec [Mikolov et al., 2013a] est une heuristique mise en œuvre avec trois couches de neurones. Il est capable de représenter un mot par un vecteur dense de taille maîtrisée. La première couche reçoit l'ensemble des textes du corpus sous forme de vecteurs. Ces vecteurs sont construits à partir de la séquence des indexes des mots. La sortie de Word2vec est un ensemble de vecteurs caractéristiques qui représentent des mots de ce corpus. Word2vec représente dans un premier temps les documents sous la forme d'une séquence puis les projette dans l'espace de description à grande dimension. Word2vec regroupe alors les vecteurs de mots similaires dans l'espace vectoriel. Les réseaux de neurones artificiels Word2vec construisent, pour chaque mot, une fenêtre de contexte. À l'intérieur de cette fenêtre, tous les mots sont traités de façon égale. Il fonctionne sans intervention humaine. En ce sens, c'est un algorithme non supervisé. Bien que Word2vec ne soit pas un réseau de neurones profond, il transforme le texte dans une représentation que les réseaux de neurones peuvent comprendre.

FIGURE III.2 – Principe de l'algorithme Word2vec (CBOW) [Rong, 2014]



Les auto-encodeurs ont pour but de reproduire leurs entrées à l'identique. Cela n'a a priori que peu d'intérêt ! Cependant, la connaissance acquise par les couches cachées à cette fin peut s'avérer très intéressante. Word2vec est similaire à un auto-encodeur, encodant chaque mot dans un vecteur. Il entraîne les mots contre d'autres mots voisins dans le corpus d'entrée.

Il le fait de deux manières : soit en utilisant le contexte pour prédire un mot cible (une méthode connue sous le nom de Continuous Bag Of Words CBOW comme décrit dans la figure III.2 , soit en utilisant un mot pour prédire un contexte cible, appelé skip-gram. La première méthode est plus rapide que la seconde mais cette dernière produit des résultats plus précis sur de grands ensembles de données.

Nous pouvons voir table III.5 la représentation continue du corpus de l'exemple 1.1. Pour ce faire nous avons entraîné le modèle Word2Vec du module gensim sur 10 époques avec une fenêtre de 5 mots et l'algorithme Skip-gram.

L'objectif de l'entraînement du modèle Skip-gram est de trouver des représentations continues de mots utiles pour prédire les mots environnants dans une phrase ou un document. Plus formellement, étant donné une séquence de mots d'entraînement $w_1, w_2, w_3, ..., w_T$ l'objectif du modèle Skip-gram est de minimiser la fonction d'erreur de log probabilité voir (équation III.7,III.8).

Équation : Word2Vec Skip-gram [Mikolov et al., 2013b]

$$-\sum_{t=1}^{T} \sum_{-m \leq j \leq m, j \neq 0} logP(w_{t+j}|w_t) \tag{III.7}$$

Avec T le nombre de mots de la séquence et m la taille du contexte.

Équation : Word2Vec CBOW [Mikolov et al., 2013b]

$$-\sum_{t=1}^{T} logP(w_t|w_{t-m}, ..., w_{t-1}, w_{t+1}, ..., w_{t+m}) \tag{III.8}$$

TABLE III.5 – Représentation continue des mots statiques : Word2Vec

Dictionnaire	index	Document 1	D 1	D 2	D 3	Document 2	D 1	D 2	D 3	Document 3	D 1	D 2	D 3
crier	1	6	-0.103	-0.1030.0456	-0.0930	7	-0.151	-0.00798	0.163	2	-0.116	-0.0674	0.0418
geek	2	2	-0.116	-0.0674	0.0418	2	-0.116	-0.0674	0.0418	1	0.131	0.125	0.117
mettre	3	9	-0.0427	0.0986	0.104	11	-0.162	0.0777	-0.0332	2	-0.116	-0.0674	0.0418
mode	4	3	-0.154	0.0245	0.144	8	-0.155	-0.0527	0.131	10	0.138	-0.0373	0.139
partir	5	12	0.157	-0.155	-0.113	0	0	0	0	0	0	0	0
pourquoi	6	4	0.0720	0.0124	0.0139	0	0	0	0	0	0	0	0
sans	7	7	-0.151	-0.00798	0.163	0	0	0	0	0	0	0	0
shell	8	13	-0.155	-0.0527	0.131	0	0	0	0	0	0	0	0
suicide	9												
url	10												
vacances	11												
vie	12												
échec	13												

Les vecteurs de mots peuvent être utilisés pour établir et comparer les associations entre mots à l'aide d'une métrique de similarité comme cosine.

1.1.4.c fastText

L'approche fastText développée par [Joulin et al., 2016] est une extension du modèle Word2vec. L'idée est d'améliorer les représentations vectorielles des mots pour les langues morphologiquement riches par une représentation vectorielle de type n-grams. Un mot est alors représenté par la somme des vecteurs de ses n-grams. Concrètement, au lieu d'apprendre en entrée directement des vecteurs de mots, fastText représente chaque mot comme un ensemble de n-grammes de caractères. Par exemple, si on prend le mot « *artificiel* » et $n = 3$, la représentation fastText de ce mot se définit par neuf tri-grams *ar, art, rti, tif, ifi, fic, ici, iel, el*. On peut ainsi obtenir une représentation pour un mot qui n'existait pas dans le corpus d'entraînement, ce qui est particulièrement intéressant pour les corpus de faible dimension.

Nous pouvons voir sur le tableau III.6 une réalisation fastText configurée sur dix epochs avec une fenêtre de rayon de trois mots pour un espace de description à trois dimensions. Ici chaque mot ou sous mot d'un document sera représenté par un vecteur de nombres réels. Un document sera représenté par une matrice à deux dimensions. La première dimension est la longueur de la séquence numérique choisie. La deuxième dimension de la matrice est le nombre de dimensions de l'espace de description des mots. Si la relation d'ordre des mots est bien respectée dans les vecteurs représentant les documents, la représentation des mots est similaire dans l'intégralité du corpus quel que soit le document. Ainsi identiquement à GloVe ou Word2vec, fastText est une méthode de représentation continue des mots statiques.

TABLE III.6 – Représentation continue des mots statiques : fastText

Dictionnaire	index	Document 1	D 1	D 2	D 3	Document 2	D 1	D 2	D 3	Document 3	D 1	D 2	D 3
crier	1	6	0.0211	-0.0238	0.0260	5	-0.0125	-0.0297	0.0458	2	-0.0241	-0.00220	-0.0246
geek	2	2	-0.0241	-0.00220	-0.0246	2	-0.0241	-0.00220	-0.0246	1	-0.00221	-0.0527	0.0835
mettre	3	9	0.00579	0.00982	0.0262	11	-0.0535	0.0243	0.0301	2	-0.0241	-0.00220	-0.0246
mode	4	3	0.00307	0.0419	-0.0484	8	-0.0445	0.0688	0.0202	10	-0.0201	-0.0440	-0.0203
partir	5	12	0.0489	-0.00413	-0.0413	0	0	0	0	0	0	0	0
pourquoi	6	4	0.00307	0.0419	-0.0484	0	0	0	0	0	0	0	0
sans	7	7	-0.00113	-0.0139	-0.0399	0	0	0	0	0	0	0	0
shell	8	13	-0.0233	0.0490	0.0148	0	0	0	0	0	0	0	0
suicide	9												
url	10												
vacances	11												
vie	12												
échec	13												

1.1.5 Représentation continue des mots contextualisés

Dans la représentation continue des mots statiques, les vecteurs de chaque mot sont construits à partir de la fenêtre de mots englobant le mot décrit. L'ordre des mots du document de la fenêtre n'est pas pris en compte. Et pourtant, il est primordial de pouvoir faire la différence entre des phrases ana-cycliques comme : « Souffrir sans amour, l'oublies-tu parfois ? » et « Parfois, tu oublies l'amour sans souffrir » . Au contraire, les représentations continues de mots contextualisées permettent de définir des vecteurs de mots prenant en compte la relation d'ordre des mots dans le document.

COVE (COntextualized word VEctors) [McCann et al., 2017] inspiré par le transfert learning sur les images, utilise un encoder de type LSTM de séquence à séquence comme pour les réseaux de neurones utilisés en traduction automatique.

UMLFIT (Universal Language Model Fine-tuning for Text Classification) [Howard and Ruder, 2018] repose sur le concept d'apprentissage par transfert à partir d'un réseau de type AWD-LSTM. L'apprentissage par transfert est l'idée que vous apprenez d'une tâche spécifique, puis que vous l'utilisez pour d'autres tâches. On peut alors utiliser de gros corpus pour faire l'apprentissage de la représentation continue des mots, puis utiliser cet apprentissage pour finaliser les modèles en vue de la classification de textes. ULMFiT a introduit une heuristique pour affiner efficacement le modèle linguistique pour diverses tâches. ULMFiT peut être appris à partir d'un grand nombre de textes bruts (comme Wikipedia) pour prédire les mots suivants. Ensuite, on peut utiliser cette connaissance apprise par transfert pour apprendre un nouveau modèle sur un ensemble de données spécifiques à une tâche, qui peut être assez petit et ainsi améliorer l'exactitude du modèle.

ELMo (Embeddings from Language Models) [Peters et al., 2018] produit une représentation de mots contextualisée profonde qui modélise à la fois (1) les caractéristiques complexes de l'utilisation des mots (par exemple, la syntaxe et la sémantique), et (2) comment ces utilisations varient selon les contextes linguistiques (c'est-à-dire pour modéliser la polysémie). Les vecteurs de description des mots sont le résultat de fonctions apprises par les états internes d'un modèle de langage bidirectionnel profond (biLM), qui est pré-entraîné sur un grand corpus de textes.

Contrairement aux descriptions vectorielles de mots traditionnelles telles que Word2vec et GLoVe, les vecteurs ELMo de représentation d'un sous mot ou d'un mot sont en fait des fonctions des phrases entières contenant ce mot. Par conséquent, le même mot peut avoir différents vecteurs de mots dans différents contextes.

Reprenons l'exemple 1.1 page 49 et considérons les documents 1 et 2.

doc 1 : Pourquoi un geek se suicide ? Pour mettre sa vie en mode sans échec.

doc 2 : Où partent les geeks en vacances ?- Aux C-Shell.

Le mot geek est généralement utilisé pour désigner une personne passionnée par un domaine. Mais sa signification peut varier suivant les communautés et regrouper différents sens comme nolifes, gamers, otaku, nerds ou encore hackers. Dans la première phrase, il pointe une personne asociale du type nolife associée à une signification négative alors que dans la deuxième phrase il désigne une personne férue d'informatique avec une sémantique plutôt positive. Il s'agit ici d'un cas de polysémie où un mot peut prendre plusieurs significations.

Les représentations ELMo sont :

— Contextuelles : la représentation de chaque mot dépend du contexte dans lequel il est utilisé.

— Profondes : les représentations de mots combinent toutes les couches d'un réseau neuronal pré-entraîné profond.

— Basées sur les caractères : les représentations sont purement basées sur les caractères, permettant au réseau d'utiliser des indices morphologiques pour former des représentations robustes pour des entités hors vocabulaire invisibles lors du pré-entraînement.

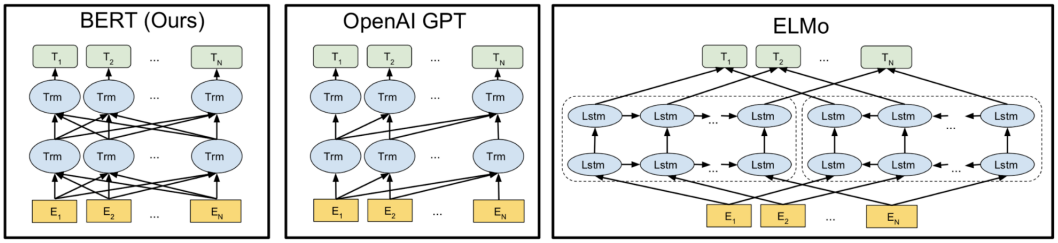
Les vecteurs de représentation des mots ELMo dépendent du contexte. Ces modèles produisent une représentation vectorielle différente pour le même mot selon le contexte dans lequel il est utilisé. C'est essentiellement pour cette raison qu'ils sur-performent les techniques précédentes dans de nombreuses tâches d'analyse automatique de textes [Peng et al., 2019].

BERT est différent d'ELMo principalement car son objectif est différent. La principale limitation d'ELMo est son incapacité à prendre en compte les contextes gauche et droit du mot cible. Même si ELMo, utilise des couches neuronales LSTM bidirectionnelles, il concatène simplement les informations de gauche à droite et de droite à gauche, ce qui signifie que la représentation ne peut pas tirer parti des contextes gauche et droit simultanément.

BERT remplace le *language modeling* ou contextualisation statistique des mots par un autre objectif appelé *Masked Language Modeling*. Dans ce modèle, les mots d'une phrase sont effacés de manière aléatoire et remplacés par un marqueur spécial « masque » avec une faible probabilité. Ensuite, un transformateur est utilisé pour générer une prédiction pour le mot masqué sur la base des mots non masqués qui l'entourent, à gauche et à droite. Grâce à ce nouvel objectif, BERT est en mesure d'atteindre des performances de pointe pour une variété de tâches comme, par exemple dans le comparatif GLUE [2] ou encore sur du question answering [Xu et al., 2019].

2. https ://gluebenchmark.com/

FIGURE III.3 – Différences entre les architectures des modèles. BERT utilise un Transformer bidirectionnel. OpenAI GPT utilise un Transformer de gauche à droite. ELMo utilise la concaténation LSTM de gauche à droite et de droite à gauche. Parmi les trois, seules les représentations de BERT sont conjointement conditionnées sur le contexte gauche et droit dans toutes les couches [Devlin et al., 2019]



BERT utilise une architecture de type Transformer, un mécanisme d'attention qui apprend les relations contextuelles entre les mots (ou sous mots) dans un texte. Le Transformer comprend deux mécanismes distincts - un encodeur qui lit l'entrée du texte et un décodeur qui produit une prédiction pour la tâche. Le but de BERT est de générer un modèle de langage, seul le mécanisme de l'encodeur est nécessaire. Le fonctionnement détaillé du Transformer est décrit dans [Devlin et al., 2019].

Contrairement aux modèles directionnels, qui lisent l'entrée de texte séquentiellement (de gauche à droite ou de droite à gauche), le codeur Transformer lit la séquence entière de mots à la fois. Par conséquent, il est considéré comme bidirectionnel, mais il serait plus exact de dire qu'il n'est pas directionnel. Cette caractéristique permet au modèle d'apprendre le contexte d'un mot en se basant sur l'ensemble de son environnement (gauche et droite du mot). BERT est pré-entraîné sur une tâche spécifique de type *next sentence prédiction*. Son vocabulaire est basé sur WordPiece [Devlin et al., 2019] utilisant le découpage de la librairie SentencePiece [Kudo and Richardson, 2018].

Dans BERT, chaque mot ou sous mot d'un document sera représenté par un vecteur de nombres réels de dimension 768. Un document sera représenté par une matrice à deux dimensions. La première dimension étant de la longueur de la séquence numérique choisie. La deuxième dimension étant du nombre de dimensions de l'espace de description des mots 768.

Nous pouvons voir dans le tableau III.7 une réalisation de la description de type BERT pré-entraîné. Si la relation d'ordre des mots est bien respectée, leur représentation est bien différenciée par document.

Nous remarquons que le vecteur de description du mot geek est découpé en deux sous mots ge et ##eek. À chaque apparition d'un de ces sous mots dans un

document les vecteurs sont différents.

À l'aide d'une mesure de similarité comme cosine, nous pouvons évaluer la différenciation du mot geek dans ces trois documents. À cette fin, nous faisons la somme des vecteurs des deux sous mots ge et ##eek pour obtenir le vecteur du mot geek. Puis nous comparons la similarité du mot dans les trois documents.

Exemple 1.2 [Similarité cosine]

$$Similarité\ cosine(geek[doc3.1], geek[doc3.2]) = 0.96$$

Dans le même document si le mot apparaît deux fois avec un contexte proche, la similarité est forte.

Exemple 1.3 [Similarité cosine]

$$Similaritécosine(geek[doc3.1], geek[doc2]) = 0.89$$

Dans deux documents séparés, le mot « geek » associé au mot « crie » dans le document trois et associé au mot « vacances » dans le document deux, la similarité est bien moins forte.

Exemple 1.4 [Similarité cosine]

$$Similarité\ cosine(geekdoc3.1, geekdoc1) = 0.94$$

Dans deux documents séparés, le mot « geek » associé au mot « crie » dans le document trois et associé au mot « suicide » dans le document un, la similarité est forte. La position du mot dans la phrase est prise en compte ainsi que le contexte sémantique du mot.

TABLE III.7 – Contextualized Word Embedding : BERT multilingual

Document 1	D 1	D 2	D 3	Document 2	D 1	D 2	D 3	Document 3	D 1	D 2	D 3
pourquoi	-0.628	3.15	-0.427	O	-1.37	-1.10	-2.39	Un	0.922	-2.90	1.54
un	0.996	-2.60	1.63	##ù	0.975	1.99	-3.40	ge	-0.617	-1.42	1.54
ge	-1.72	-0.759	0.307	parte	-1.10	1.19	-1.88	##ek	-0.0849	-1.98	2.33
##ek	-0.167	-1.05	1.60	##nt	0.527	3.42	-3.28	ne	2.22	3.24	-3.97
se	2.06	-1.02	-1.63	les	-0.468	-1.27	-1.77	c	2.58	1.02	-1.74
suicide	0.995	0.705	-3.40	ge	-0.361	-0.715	-0.198	##rie	3.29	-0.334	-1.14
?	-1.56	2.37	-1.23	##eks	1.50	-2.46	0.933	pas	3.74	-0.868	-2.43
Pour	0.253	0.0272	0.126	en	3.46	-0.746	-1.21	,	-0.667	0.222	-3.26
mettre	0.674	-0.738	0.694	vacances	5.02	-1.17	-1.61	un	-1.85	-3.18	-0.734
sa	2.70	-1.35	-0.649	?	1.16	1.92	-2.089	ge	-1.55	-0.954	-0.0948
vie	3.51	-1.22	-2.36	Aux	-0.0668	0.953	-2.55	##ek	-0.731	-0.210	-0.208
en	2.19	-2.08	-1.50	C	1.31	1.29	-0.109	URL	3.48	0.491	-1.61
mode	1.77	0.527	-3.60	-	3.60	1.58	-0.0804	.	-0.339	0.0342	-1.04
sans	4.20	-2.31	-4.94	Shell	-0.425	0.578	-0.203		0	0	0
échec	3.26	-1.92	-4.10	.	-1.64	-1.38	0.0333		0	0	0
.	-0.524	0.326	-2.59		0	0	0		0	0	0

Le vecteur de description d'un mot sera unique pour un mot donné dans un document à une certaine position. La description continue des mots peut être

statique ou dynamique suivant que les poids de la couche de neurones sont fixés au départ de l'entraînement du réseau ou imprégnés par la rétro-propagation du gradient durant la phase d'entraînement.

1.2 Classifieurs statistiques

Les modèles utilisés dans cette thèse sont résumés dans le tableau III.8. Les modèles d'apprentissage classiques comparés sont de type SVM (SVC), Naïve Bayes (CNB), Arbre de décision (DT, RF), boosting (ADB,Xgboost), K-plus-proches-voisins (KNC). Nous en faisons une description succincte dans cette section.

TABLE III.8 – Modèles de Classification : statistique

KNeighborsClassifier (KNN)	Abbasifard et al. [2014]
Complement Naive Bayes (CNB)	Rennie et al. [2003]
Decision Trees (DT)	Breiman et al. [1984]
Random Forest (RF)	Breiman [2001]
AdaBoost (AB)	Freund and Schapire [1997]
XGBoost (XGB)	Chen and Guestrin [2016]
LinearSVC (SVM)	Wu et al. [2004]
Méta Classifier Commettee (MCC)	

1.2.1 Bayes naïf

Complémentaire Bayes Naïf (CNB) est une adaptation de l'algorithme Bayes naïf multinomial standard qui est particulièrement adapté aux ensembles de données déséquilibrés. Plus précisément, il utilise les statistiques sur chaque classe pour évaluer les poids du modèle.

1.2.2 KNeighborsClassifier (KNN)

Ce classifieur ne procède pas à un apprentissage. KNN stocke simplement les instances des données. La classification est calculée à partir d'un vote à la majorité simple des voisins les plus proches de chaque point. Un point se voit attribuer la classe de données qui a le plus de représentants parmi les voisins les plus proches de lui-même. La distance se calcule suivant l'espace de représentation : Cosine, Euclidienne ou Manhattan, etc.

1.2.3 Decision Trees (DT)

Les arbres de décision sont des classifieurs qui basent leurs décisions sur une suite de tests associés aux caractéristiques des données. À un arbre de décision,

est associé simplement une procédure de classification. À chaque description complète d'une donnée est associée une et une seule feuille de l'arbre de décision. Cette association est définie en commençant à la racine de l'arbre et en descendant dans l'arbre selon les réponses aux tests qui étiquettent les nœuds internes. La classe associée est alors la classe majoritaire associée à la feuille qui correspond à la description.

1.2.4 Random Forest (RF)

RF est un méta-classifieur construit sur un ensemble diversifié de classifieurs de type « arbre de décision » . Il est créé en introduisant un caractère aléatoire dans la construction de l'ensemble des arbres de décision par choix aléatoire des descripteurs et échantillonnage. La prédiction de l'ensemble est donnée comme la prédiction moyenne des classificateurs individuels.

1.2.5 AdaBoost (AB)

Le principe de base d'AdaBoost est d'adapter une séquence d'apprenants faibles (c'est-à-dire des modèles qui ne sont que légèrement meilleurs que les suppositions aléatoires, tels que de petits arbres de décision) sur des versions des données modifiées à plusieurs reprises. Les prédictions de chacun d'eux sont ensuite combinées par le biais d'un vote majoritaire pondéré (ou somme) pour produire la prédiction finale.

1.2.6 XGBoost (XGB)

Il construit ses modèles à partir de modèles a priori faibles de manière itérative. En boosting, les modèles individuels ne sont pas construits sur des sous-ensembles de données et de fonctionnalités complètement aléatoires comme les RF mais séquentiellement en mettant plus de poids sur les instances avec de mauvaises prédictions et des erreurs élevées. L'idée générale derrière cela est la focalisation du modèle vers les cas difficiles à prédire correctement de sorte que le modèle apprenne des erreurs passées. Le gradient est utilisé pour minimiser une fonction de perte, semblable à la façon dont les réseaux neuronaux utilisent la descente du gradient pour optimiser les poids. À chaque cycle durant la phase d'entraînement, l'apprenant faible est construit et ses prédictions sont comparées aux résultats corrects attendus.

1.2.7 LinearSVC (SVM)

Le classifieur SVM ou Séparateur à Vaste Marge repose sur l'application d'algorithmes de recherche de règles de décision linéaires « hyperplan sépara-

teur » l'idée est de rechercher une séparation maximale des données par classe. La recherche de cette limite de séparation s'effectue toutefois dans un espace de très grande dimension, lequel est l'image de l'espace d'entrée original par une transformation dénommée noyau.

1.2.8 Méta Classifier Commettee (MCC)

Ce classifieur est une agrégation des classifieurs précédents par « Hard votting » . Nous construisons ce modèle en utilisant les modèles précédents. Chaque modèle est entraîné séparément sur les textes de l'ensemble d'apprentissage. Nous construisons un comité de modèles ou chaque modèle aura un droit de vote lors de la prédiction de la classe d'un texte. En vue d'exprimer le vote d'un modèle, nous requêtons la classe qu'il prédit pour un texte particulier. Le Hard voting consiste alors à proposer le choix de la classe majoritaire lors des votes des modèles du comité.

1.3 Classifieurs basés sur des modèles neuronaux convolutionnels

Dans ce qui suit, nous discutons plus précisément les modèles d'apprentissage profond de type CNN. Ils utilisent une numérisation des mots de type séquence vectorielle suivie par une couche de représentation continue des mots.

TABLE III.9 – Modèles de Classification : apprentissage profond CNN

Convolutional Neural Network (CNN)	Johnson and Zhang [2015]
Multi-Group Norm Constraint Convolutional Neural Network(MGNCCNN)	Zhang et al. [2016b]
Convolutional Neural Network 2 Dimensions (CNN2D)	Kim [2014]

1.3.1 Convolutional Neural Network (CNN)

Ils sont structurés par deux opérations : convolution puis max-pooling voir figure 6 page 37. La convolution est basée sur plusieurs filtres combinés pour extraire les nombreuses propriétés associées aux données. La seconde opération compresse les résultats de l'opération précédente pour extraire des informations denses.

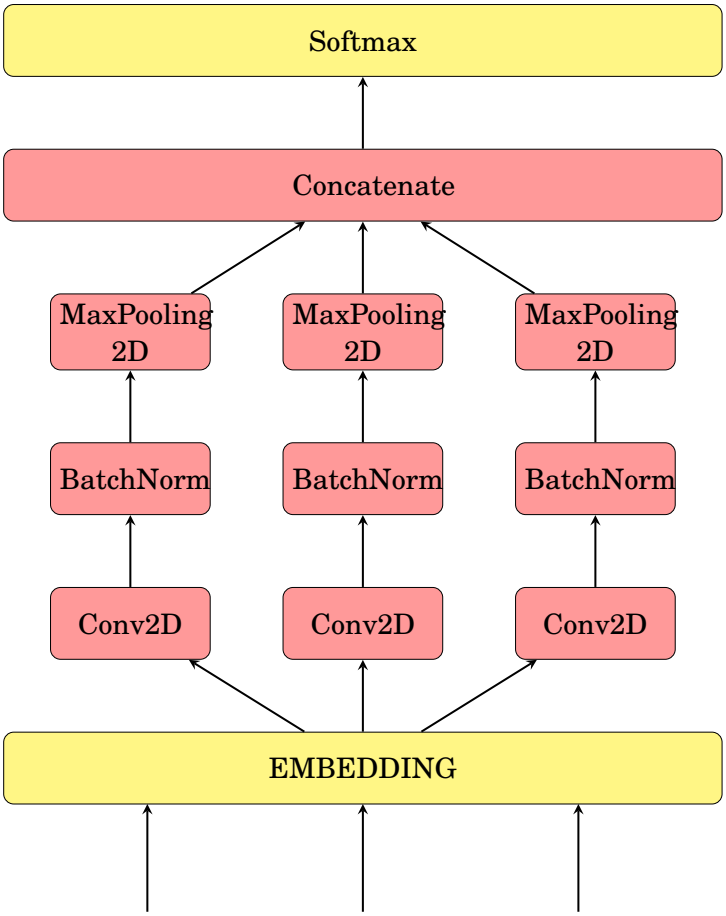


FIGURE III.4 – Réseau CNN

1.3.2 Multi-Group Norm Constraint Convolutional Neural Network (MGNCCNN)

Il capitalise sur plusieurs ensembles de description de mots pour la classification des phrases. MGNCCNN utilise plusieurs couches d'embedding de dimensions différentes, puis il fait une jointure de celles-ci avant une couche de convolution à une dimension pour former enfin le vecteur final.

1.3.3 Convolutional Neural Network 2 Dimensions (CNN2D)

CNN2D est une architecture permettant l'utilisation des vecteurs de description des mots suivie de trois couches de convolution en deux dimensions sur plusieurs canaux.

1.4 Classifieurs basés sur des modèles neuronaux récurrents

Dans ce qui suit, nous discutons plus précisément les modèles d'apprentissage profonds de type LSTM. Tous nos modèles sont construits à partir de la même couche d'entrée et de la même couche de sortie. La couche d'entrée est une couche de type embedding permettant de caractériser chaque mot du document analysé par un vecteur de dimension 300. La couche de sortie contient-elle, autant de neurones qu'il y a de classes dans notre ensemble de données ? Ensuite, nous décrivons les couches intermédiaires de nos différents modèles de réseaux neuronaux. Nous invitons le lecteur à consulter des revues plus détaillées telles que celle de Schmidhuber [2014].

TABLE III.10 – Modèles de Classification : apprentissage profond LSTM

Long short-term memory (LSTM)	[Hochreiter and Schmidhuber, 1997]
Convolution Long short-term memory (CLSTM)	[Zhou et al., 2015]
Asymmetric Convolutional Bidirectional Long short-term memory (ACLSTM)	[Liang and Zhang, 2016]
Bidirectional Long short-term memory (BLSTM)	[Song et al., 2018]
Attention Bidirectional Long short-term memory (ABLSTM)	[Vaswani et al., 2017b]

1.4.1 Long short-term memory (LSTM) RNN

LSTM est un sous-type de RNN. Les réseaux neuronaux récurrents permettent l'étude de séquences de données. Ces réseaux, composés de plusieurs couches, estiment leurs sorties en fonction des états de leurs couches précédentes en utilisant une mémoire intermédiaire. LSTM est basé sur des blocs de mémoire qui sont utilisés comme unités dans la couche récurrente pour capturer des dépendances à plus longue portée.

1.4.2 Convolution Long short-term memory (CLSTM)

Le modèle extrait des corrélations à partir de l'espace de description des mots grâce à la couche de convolution et l'information est alors transmise sous forme de séquence à la couche LSTM suivante. La sortie de chaque convolution contient les informations sémantiques des phrases entières sous forme de séquence.

1.4.3 Asymmetric Convolutional Bidirectional LSTM (ACLSTM)

Les auteurs [Liang and Zhang, 2016] proposent un modèle qui combine CNN asymétrique et LSTM bidirectionnel. Dans le but de rendre les modèles profonds, ils ont factorisé les convolutions ayant différentes tailles de filtre.

1.4.4 Bidirectional Long Short-Term Memory (BLSTM)

BLSTM permet d'utiliser l'information après et avant les données étudiées par le réseau au temps t. Une architecture de mise en œuvre est représentée table III.5.

1.5 Classifieurs basés sur des modèles neuronaux contextuels dynamiques

Dans cette section, nous présentons les classifieurs basés sur des modèles neuronaux contextuels dynamiques répertoriés dans la table III.11. Ces modèles sont des extensions du modèle BERT (Bidirectional Encoder Representations from Transformers). La description de ce modèle a été faite à la sous-section 1.1.5.

1.5.1 Distillation Bidirectional Encoder Representations from Transformers (DistilBERT)

Il exploite la distillation de l'information pendant la phase de pré-entraînement du modèle. La distillation des connaissances [Hinton et al., 2015] est

TABLE III.11 – Modèles de Classification : apprentissage profond contextuel

Bidirectional Encoder Representations from Transformers (BERT)	[Devlin et al., 2019]
A distilled version of BERT (DistilBERT)	[Sanh et al., 2019]
A Lite BERT for Self-supervised Learning of Language Representations (ALBERT)	[Lan et al., 2019]
A Pretrained Language Model for Scientific Text (SciBERT)	[Beltagy et al., 2019]
A Robustly Optimized BERT Pretraining Approach (RoBERTa)	[Liu et al., 2019]
Generalized Autoregressive Pretraining for Language Understanding (XLNet)	[Yang et al., 2019]

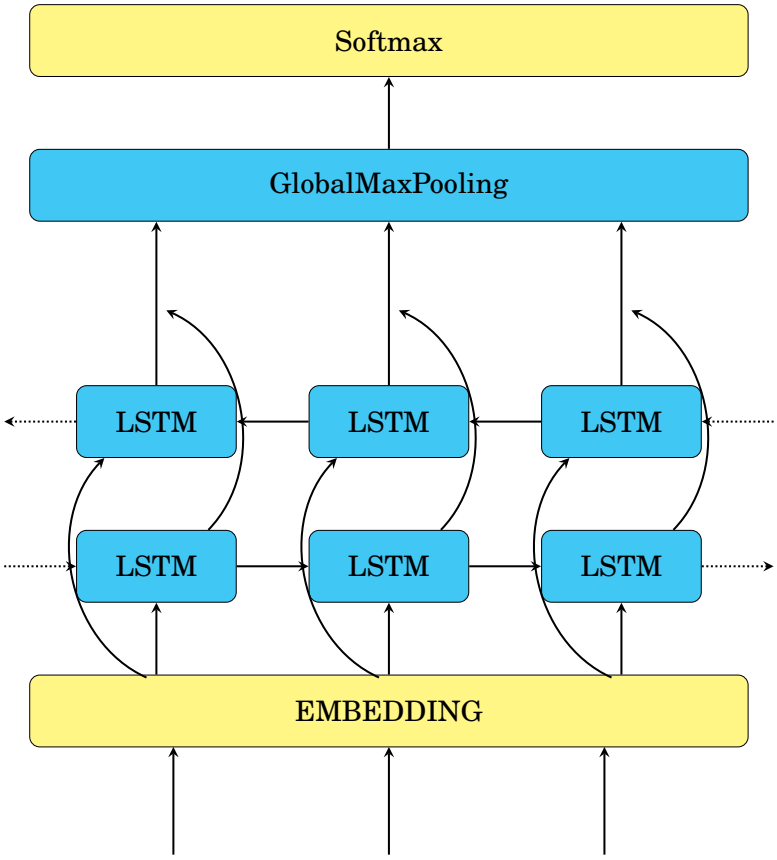


FIGURE III.5 – Réseau LSTM

une technique de compression dans laquelle un modèle compact « l'élève » est formé pour reproduire le comportement d'un modèle plus grand nommé « l'enseignant » . Il est alors possible de réduire la taille du modèle BERT de 40%, tout en conservant 97% de ses capacités de compréhension du langage et tout en étant 60% plus rapide. Pour tirer parti des biais inductifs appris par les modèles plus grands lors du pré-entraînement, DistilBERT utilise une triple fonction d'erreur combinant modélisation du langage, distillation avec supervision et distance cosine entre l'élève et l'enseignant.

1.5.2 A Lite BERT for Self-supervised Learning of Language Representations (ALBERT)

ALBERT intègre deux techniques de réduction des paramètres qui lèvent les principaux obstacles à la mise à l'échelle des modèles pré-entraînés. La première est une factorisation des poids de la description des mots. Il décompose cette matrice de description des mots en deux petites matrices. Cette séparation permet de limiter la taille des couches cachées sans augmenter de manière significative la taille du dictionnaire de description des mots. La deuxième technique est le partage de paramètres entre les couches. Cette technique empêche les paramètres de croître avec la profondeur du réseau. Les deux techniques réduisent considérablement le nombre de paramètres par rapport à BERT sans nuire gravement aux performances. Pour améliorer encore les performances d'ALBERT, les auteurs introduisent également une fonction d'erreur sur la phase de pré-entraînement dénommée SOP (Sentence-Order Prediction) en place de NSP (Next Sentence Prédiction) proposée avec le réseau BERT original.

1.5.3 A Pretrained Language Model for Scientific Text (SciBERT)

SciBERT utilise la même architecture que BERT mais est, à la place, préentraîné sur un ensemble de textes scientifiques. De plus, il est construit sur la base de SCIVOCAB, un nouveau vocabulaire WordPiece est introduit sur le corpus scientifique utilisant le découpage de la bibliothèque SentencePiece.

1.5.4 A Robustly Optimized BERT Pretraining Approach (RoBERTa)

RoBERTa améliore les performances de BERT par un pré-entraînement plus exigeant. Le pré-entraînement du modèle est plus long, avec des batchs plus importants, sur plus de données. La tâche de prédiction de la prochaine phrase (next sentence prédiction) n'est plus utilisée. Le pré-entraînement se fait sur des séquences plus longues. Le masque sur les données d'apprentissage est modifié dynamiquement. Avec également, un nouvel ensemble de données de préentraînement plus volumineux (CC-NEWS).

1.5.5 A Generalized Autoregressive Pretraining for Language Understanding (XLNet)

XLNet est une méthode de pré-entraînement autorégressive qui permet d'apprendre des contextes bidirectionnels, en maximisant la probabilité attendue sur toutes les permutations d'ordre des séquences de mots.

1.6 Conclusion

L'ensemble des techniques présentées précédemment est synthétisé dans la figure III.6

Cette section a permis de montrer le foisonnement des approches dans le domaine de la classification de textes.

Dans la section suivante 2, nous allons présenter la méthode utilisée pour comparer ces différentes approches sur les jeux de données présentés dans le chapitre I. Puis nous présenterons, dans la section 3, les résultats de cette comparaison.

2 Méthodologie de comparaison des différents classifieurs de textes

Nous présentons ici la méthode utilisée en vue de réaliser la comparaison des divers modèles mis en œuvre lors des expérimentations que nous avons conduites.

2.1 Entrées et sorties des modèles

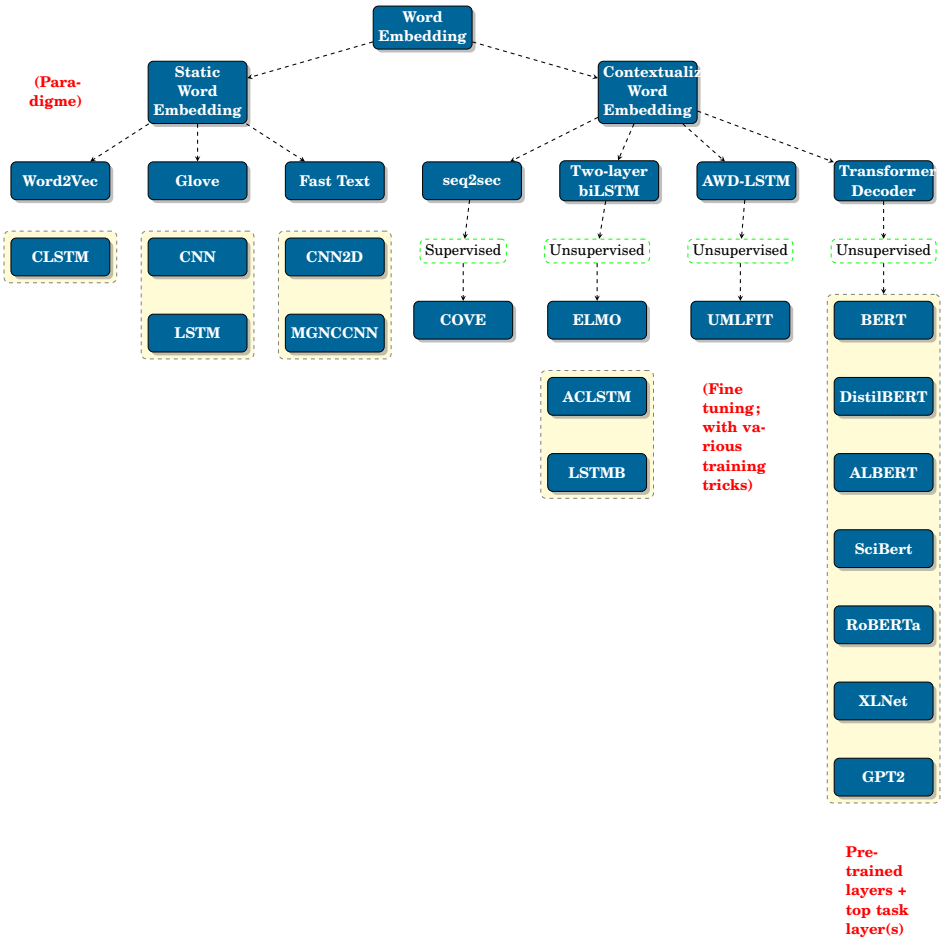
Les entrées des modèles de tous les classifieurs sont les documents.

Pour les modèles statistiques, nous utilisons pour le pré-traitement de chaque classifieur, une vectorisation basée sur la mesure TF-IDF.

Pour les méthodes d'apprentissage profond de type CNN et LSTM, les données sont préparées différemment. Nous conservons les 15 000 mots les plus fréquents (hors stop words) et nous représentons chaque document par une séquence de mots ce qui permet de conserver l'ordre des mots. Tous nos modèles de type LSTM ou CNN sont construits à partir d'une couche d'entrée de type embedding de dimension 300 et de la même couche de sortie. La couche d'entrée permet de caractériser chaque mot du document.

Ensuite, nous décrivons les couches intermédiaires de nos différents modèles de réseaux neuronaux. Nous invitons le lecteur à consulter des revues plus détaillées telles que celle de Schmidhuber [2014].

FIGURE III.6 – Synthèse des techniques de représentation numérique de documents concernant les réseaux de neurones.



Pour les méthodes d'apprentissage profond de type BERT, les données sont préparées différemment. Nous conservons l'ensemble des mots (hors stop words), chaque modèle ayant son propre dictionnaire de sous-mots.

La couche de sortie, quant à elle, contient autant de neurones qu'il y a de classes dans notre ensemble de données.

2.2 Partitionnement des données et entraînement

Dans le but d'évaluer les différents modèles, nous appliquons une validation croisée en quatre plis. L'ensemble de données est divisé en quatre sous-ensembles. Nous utilisons trois sous-ensembles pour la phase d'apprentissage et un sous-ensemble pour la phase de test. Nous avons répété ce processus quatre fois. Pour chaque pli, nous utilisons des ensembles d'apprentissage et de test différents. Puis nous calculons une métrique pour évaluer la performance.

Pour les algorithmes de classification classiques nous utilisons l'outil sklearn [3] avec les paramètres par défaut.

Pour toutes les architectures d'apprentissage profond de type CNN ou LSTM, nous avons utilisé une taille de 32 pour les lots (c'est-à-dire : le nombre d'instances d'entraînement à considérer en même temps), la dimension de représentation continue des mots (chaque mot est décrit par un vecteur de dimension 300), le nombre d'epoque 25 (le nombre d'itérations sur l'ensemble d'entraînement), la fonction d'activation de type Selu [Göçeri, 2019] et le taux d'abandon de neurones 0.3 (Dropout ratio, ratio d'unités cachées à désactiver dans chaque formation des lots).

Pour les modèles de type BERT, la taille des lots de données d'entraînement est fixée à 16 pour des limitations de taille de mémoire. L'ensemble des hyperparamètres des réseaux est fixé pour l'ensemble des jeux de données. Le taux d'apprentissage est fixé à 0.00002. Tous nos réseaux de neurones sont entraînés avec un processus de back-propagation utilisant une fonction d'erreur (*loss*) de type entropie croisée. Les optimiseurs de calcul du gradient sont de type *Adam weight*. De plus, nous utilisons une mise à jour du taux d'apprentissage linéaire avec une amélioration du départ. Les expérimentations sont faites sur deux GPU de type GeForce RTX.

2.3 Mesure de la performance

Pour tous les modèles, nous nous sommes basés sur une métrique couramment utilisée dans le domaine de la classification. Chaque sortie est un vecteur

3. http://scikit-learn.org/stable/

de dimension n correspondant aux n classes à prédire. Nous mesurons la qualité des classifieurs avec la métrique *exactitude* (accuracy). Celle-ci exprime le nombre de classes prédites justes en regard du nombre total de prédictions de classes réalisées. Nous avons choisi d'utiliser une métrique stricte en considérant une sortie comme juste uniquement si l'intégralité des classes prédites sont exactes.

Exemple 2.1 [Exactitude]

Considérons 3 documents décrits par un vecteur d'annotations pour 4 classes. Chaque élément i de ce vecteur correspond à la présence (1) ou à l'absence (0) de la classe i pour le document.

$Document1 \longrightarrow 0010$

$Document2 \longrightarrow 1000$

$Document3 \longrightarrow 0100$

Considérons maintenant un modèle retournant 3 vecteurs de prédictions de ces 4 classes. Chaque élément i de ce vecteur correspond à la prédiction (1) ou à l'absence de prédiction (0) de la classe i pour le document.

$prédiction1 \longrightarrow 0010$

$prédiction2 \longrightarrow 0010$

$prédiction3 \longrightarrow 1100$

Seule la première prédiction sera considérée comme juste pour les trois prédictions émises. L'exactitude du modèle sera égale à : $exactitude = \frac{1}{3} = 0.333$

TABLE III.12 – Matrice de confusion.

	Appartient à la classe	N'appartient pas la classe
test positif	Vrai Positif	Faux Positif
test négatif	Faux Négatif	Vrai Négatif

Équation : Exactitude

$$Exactitude = \frac{VP + VN}{VP + VN + FP + FN} \tag{III.9}$$

Dans le cas où le nombre échantillons par classe est différent, un classifieur qui en phase de test propose systématiquement la classe majoritaire pourrait avoir une exactitude de valeur élevée.

Exemple 2.2 [Exactitude d'un classifieur d'une classification déséquilibrée]

Prenons le cas d'une classification binaire. La première classe a 75 000 individus tandis que la deuxième classe a 10 000 individus.

$Exactitude = \frac{75000}{75000+10000} = 0.8823$

Le classifieur proposant toujours la classe majoritaire aura une exactitude de valeur 0,882 alors qu'il est trivial à réaliser.

En conséquence, on peut alors en tenir compte dans les commentaires afin de ne pas surestimer la performance du classifieur, ou alors, réaliser l'évaluation avec d'autres métriques.

La précision mesure le taux de prédictions exactes de chaque classe en rapport au nombre de prédictions émises sur chaque classe.

Équation : Précision

$$Précision = \frac{VP}{VP+FP} \tag{III.10}$$

Le rappel exprime l'aptitude du classifieur à retrouver à travers ses prédictions tous les échantillons d'une classe.

Équation : Rappel

$$Rappel = \frac{VP}{VP+FN} \tag{III.11}$$

La f-mesure est la moyenne harmonique de la précision et du rappel.

Équation : F-mesure

$$F-mesure = 2 \times \frac{Précision \times Rappel}{Précision + Rappel} \tag{III.12}$$

Kappa de Cohen est une métrique statistique qui mesure l'accord inter-annotateur ou classifieur. Cette mesure donne un score qui exprime le niveau d'accord entre deux annotateurs sur un problème de classification. Il est défini comme :

Équation : Kappa de Cohen

$$k = \frac{p_0 - p_e}{1 - p_e} \tag{III.13}$$

Où p_0 est la probabilité empirique d'accord sur l'étiquette attribuée à n'importe quel échantillon, et p_e est l'accord attendu lorsque les deux annotateurs attribuent des étiquettes au hasard. Cette métrique est estimée à l'aide d'un a priori empirique par annotateur sur les étiquettes de classe. On mesure ici la performance du classifieur étudié en vis-à-vis avec un classifieur qui choisirait uniquement la classe majoritaire.

Kappa de Fleiss fonctionne pour n'importe quel nombre d'annotateurs ou classifieurs contrairement au Kappa de Cohen qui est limité à deux annotateurs.

Équation : Kappa de Fleiss

$$k = \frac{\overline{P} - \overline{P_e}}{1 - \overline{P_e}} \tag{III.14}$$

Où $\overline{P}$ est moyenne des probablités sur les échantillons et $\overline{P_e}$ la moyenne des probabilités sur les classes en vis à vis des classifieurs.

Dans le cadre d'une classification, on mesurera l'écart avec le choix d'une quelconque classe au hasard.

La courbe ROC est la courbe des faux positifs en fonction des vrais positifs et n'est pas sensible aux déséquilibres de classes.

On peut également, pour tenir compte du déséquilibre de classes modifier la répartition des échantillons par classe lors de la phase d'entraînement par sous échantillonnage (Undersampling) ou sur-échantillonnage (Oversampling).

Le sous-échantillonnage se faisant en diminuant le nombre d'échantillons des classes majoritaires vers l'effectif de la classe minoritaire.

Le sur-échantillonnage se faisant en augmentant de manière aléatoire les effectifs des classes minoritaire vers la taille de la classe majoritaire.

Enfin pour débiaiser le modèle vers la classe minoritaire, on peut également appliquer un poids à chaque prédiction d'étiquette inversement proportionnel au déséquilibre de classe. Cette technique est appelée le cost-sensitive learning et a été proposée par Elkan [2001].

3 Résultats des expérimentations

Dans cette section, nous présentons les résultats des différents modèles présentés dans les sections 1.2, 1.3, 1.4, 1.5 appliqués aux jeux de données décris dans la section 3 page 22.

3.1 Comparaison des modèles de classification

Le tableau III.13 synthétise les résultats des différents classifieurs statistique sur les jeux de données. Comme l'on pouvait s'y attendre, les résultats pour les jeux avec peu de classes sont supérieurs aux résultats pour des jeux avec beaucoup de classes à l'exception du jeu COVID-19 qui obtient les meilleurs résultats d'exactitude dans l'absolu. Très clairement les méta-classifieurs comme

TABLE III.13 – Comparaison des performances des classifieurs statistiques.

jeux	Drugs.com	eRisk anorexie	eRisk dépression	PubMed RCT	COVID 19	SST
KNC	0.2805	0.9181	0.9037	0.5167	0.9492	0.4998
CNB	0.3183	0.8423	0.8440	0.6221	0.9338	0.305
Decision-Tree	0.1146	0.8333	0.8451	0.3711	0.8565	0.4883
Rand-Forest	0.3368	0.9138	0.8868	0.6212	0.9342	0.5082
AdaBoost	0.3174	0.9032	0.8867	0.6225	0.9531	0.498
XGBoost	0.3157	0.9164	0.9032	0.6136	**0.9573**	0.496
SVC	0.326	0.9127	0.8899	0.6410	0.9510	0.4788
MCC	**0.3458**	**0.9185**	**0.9054**	**0.6520**	0.9488	**0.51**

XGBoost ou randForest surperforment les simples classifieurs comme Décision Tree. Le classifieur en comité regroupant quant à lui les autres classifieurs tire parti de sa pluralité et sur-performe pour cinq jeux sur six.

Dans la table III.14, nous retrouvons les résultats pour les classifieurs basés sur les réseaux de neurones classiques. Ces classifeurs obtiennent des résultats comparables aux classifieurs statistiques même si leurs résultats sur la métrique d'exactitude est inférieure pour le meilleur de leur catégorie en comparaison avec le meilleur classifieur statistique pour quatre jeux sur six. Par exemple, pour la classification du jeu Drugs.com, le meilleur classifieur traditionnel obtient une exactitude de 0.3458 alors que le meilleur classifieur en apprentissage profond obtient une exactitude de 0.3246.

Les résultats des classifieurs neuronaux de dernière génération sont reportés table III.15. Pour chaque type de classification, les meilleurs classifieurs avec des technologies utilisant une architecture à Transformer ont obtenu des résultats supérieurs aux autres types de classifieurs. L'écart peut néanmoins être plus ou moins important. Par exemple, pour le jeu eRisk anorexie, l'écart entre MCC 0.9185 et BERT 0.9239 est de 0.0054. Pour le jeu PubMed RCT, l'écart entre MCC 0.6520 et XlNet 0.7423 est de 0.0903, ce qui est important.

La comparaison des matrices de confusion présentes dans la figure III.7 montre que si les résultats avec la métrique d'exactitude sont proches, l'amélioration apportée par les classifieurs neuronaux de type BERT se retrouvent pour l'ensemble des termes de la matrice de confusion.

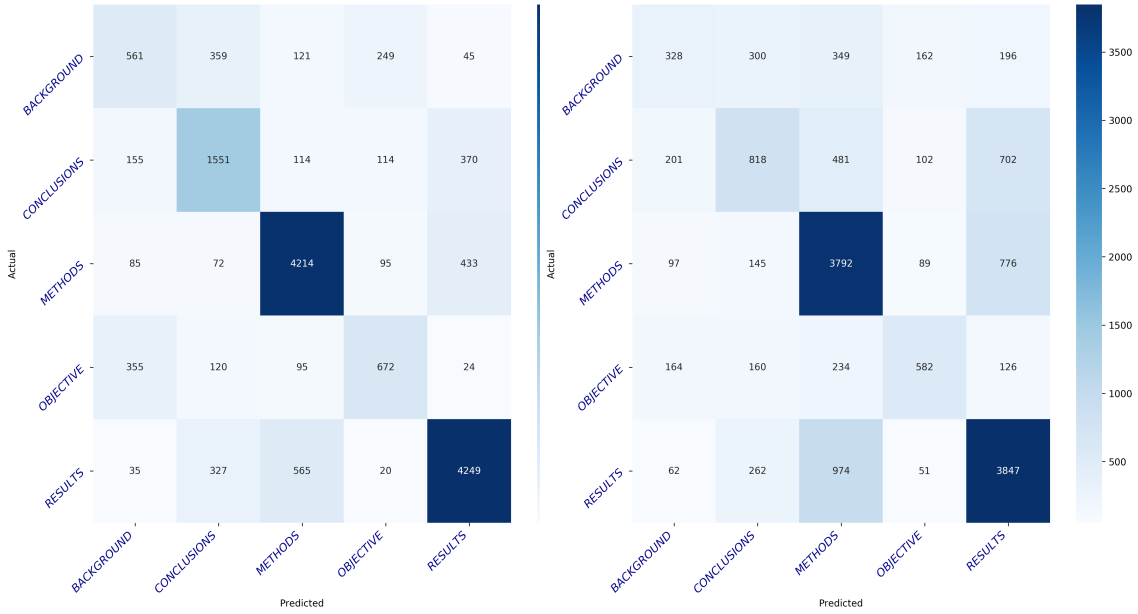
TABLE III.14 – Comparaison des performances des classifieurs neuronaux de première génération.

jeux	Drugs.com	eRisk anorexie	eRisk dépression	PubMed RCT	COVID 19	SST
MLP + HV	0.2034	0.9122	0.9068	0.3452	0.9342	0.3176
CNN + GloVe	0.2645	0.9113	0.9068	0.4098	0.9342	0.4885
CNN2D+-FastText	0.3137	0.9112	0.9056	0.5828	0.9342	0.4996
MGNCCNN-+FastText	0.3164	0.9122	0.9068	0.526	0.9342	0.4994
CLSTM + Word2vec	0.3164	0.9122	**0.9068**	0.526	0.9342	0.5002
LSTM + GloVe	0.3153	0.9122	0.9068	0.5539	0.9342	0.5002
ACLSTM + ELMO	0.2054	0.9122	0.9067	0.6367	0.4784	0.5150
LSTMB + ELMO	**0.3246**	**0.9165**	0.897	**0.6888**	**0.9529**	**0.5486**

TABLE III.15 – Comparaison des performances des classifieurs neuronaux de deuxième génération.

jeux	Drugs.com	eRisk anorexie	eRisk dépression	PubMed RCT	COVID 19	SST
BERT	0.3423	0.9188	**0.9067**	0.7312	0.9568	0.5856
DistilBERT	0.3420	0.9208	0.9054	0.7156	0.9524	**0.5948**
AlBERT	0.3411	0.9230	0.8980	0.7135	0.9505	0.5658
ScienceBERT	0.3564	0.9228	0.8915	-	0.9538	0.5241
RoBERTa	0.3671	0.9205	0.897	0.729	0.9562	0.5788
XLNet	**0.3678**	**0.9239**	0.9019	**0.7423**	**0.9571**	0.5442

FIGURE III.7 – Matrice de confusion de la classification avec le jeu de données PubMed RCT pour le classifieur RoBERTa à gauche et pour le classifieur AdaBoost à droite.



3.2 Limites de l'étude

Notre approche préliminaire présente de nombreuses limites que nous détaillons dans cette section.

Concernant la phase de classification, la principale limite est que nous avons utilisé uniquement l'exactitude comme métrique pour évaluer la performance des classifieurs. Le rappel et la précision sont deux mesures importantes pour évaluer la qualité d'un classifieur et devrait être prises en considération. Une autre limitation porte sur l'interprétation de l'apprentissage profond. Ces modèles sont des "boîtes noires". Ils ne fournissent pas d'explication même si la prédiction est efficace [Shwartz-Ziv and Tishby, 2017]. Cependant, malgré un nombre de messages manuellement annotés limités, cette étude a montré que la performance des modèles traditionnels était plus faible que la performance des meilleurs classifieurs profonds récents. De nouvelles techniques d'apprentissage profond sont étudiées pour faciliter l'interprétation de tels modèles [Liu et al., 2017; Lipton, 2016] et pourraient s'avérer judicieuses dans ce contexte.

Pour finir, une dernière limite de notre étude est liée à la généralisation de nos résultats. En effet, la tâche est très spécifique au sujet d'étude, les petits jeux de données ainsi que le type de textes rendent difficile la généralisation de notre approche. Nos résultats montrent une variabilité dans la qualité de classification

suivant les jeux de données. Cela tendrait à dire que l'analyse y est spécifique et en conséquence que les réseaux ne font que s'adapter sans généraliser leurs connaissances.

Toutefois, notre méthodologie et les résultats peuvent être utilisés comme référence pour d'autres études sur l'identification automatique de catégories à partir de données textuelles.

4 Conclusion

Notre étude a souligné l'efficacité des architectures d'apprentissage profond pour prédire des catégories sur des données de forum, des textes scientifiques, des commentaires d'utilisateur. La faisabilité de notre approche pourrait conduire à de nouvelles applications en santé basées sur les médias sociaux destinés aux patients et aux professionnels de santé.

Premièrement, nous proposons l'utilisation de l'apprentissage automatique et de la visualisation interactive dans l'intention de donner sens aux prédictions [Mercadier et al., 2018] pour relever ces défis. Cette étude reste préliminaire. Une étude plus approfondie sur les différents types de préparation des données, paramètres des algorithmes, modèles d'apprentissage permettrait d'affiner l'interprétation des résultats de la phase d'apprentissage et de test.

Deuxièmement, nous pensons que lorsque les ensembles de données sont petits, l'apprentissage est difficile. Une amélioration significative serait la mise en œuvre de techniques d'apprentissage actif que nous nous proposons d'étudier dans le chapitre suivant. En effet, dans ce type de tâche, il est important d'optimiser les informations disponibles afin que les systèmes de classification puissent les utiliser le plus efficacement possible pendant la phase d'apprentissage tout en préservant l'acquisition de nouveaux échantillons étiquetés [Ducoffe and Precioso, 2015].

IV

Apprentissage profond actif

Plan du chapitre

Récemment, la classification de documents textuels a beaucoup progressé. Cependant, les modèles utilisés doivent généralement s'entraîner au préalable avec de nombreux échantillons étiquetés. Il est, semble-t-il, possible de diminuer ce nombre d'échantillons en choisissant mieux les données à annoter via des techniques d'apprentissage actif. Cela peut permettre de diminuer le coût du processus en réduisant l'intervention humaine.

Dans ce chapitre, nous adapterons les techniques récentes d'apprentissage actif profond utilisées pour la classification d'images, au cas de l'analyse de textes.

En particulier, nous serons attentifs à l'apport de l'apprentissage actif profond selon les modèles récents d'analyse de textes utilisée comme BERT ou XLNet. Nous validerons nos hypothèses sur des jeux de données de notre étude.

Le chapitre s'articule en cinq sections : une section introduction 1 suivie d'un état de l'art dans la section 2, la méthode décrite dans la section 3, les expérimentations dans la section 4, les résultats expérimentaux dans la section 5 et enfin une conclusion dans la section 6.

1 Introduction

Une des difficultés majeures lors de l'exploration de données par des méthodes d'apprentissage supervisées est de posséder un jeu de données suffisant pour l'entraînement des modèles. En effet, il est nécessaire de catégoriser les données manuellement avant de réaliser l'étape d'apprentissage. La taille des jeux de données rend cette tâche de catégorisation très coûteuse. Il convient de réduire ce coût par des systèmes semi-automatiques.

Dans ce contexte, l'apprentissage actif, pendant lequel l'oracle intervient pour choisir les exemples à étiqueter, s'avère prometteur. L'intuition est la suivante : en choisissant les exemples intelligemment et non aléatoirement, les modèles devraient s'améliorer avec moins d'efforts et donc à moindre coût (c'est-à-dire avec moins d'exemples annotés).

Dans ce chapitre, nous conduisons une étude dans l'intention d'évaluer la qualité des processus d'apprentissage actif pour une tâche spécifique de la classification multi-classes de textes. Cette évaluation doit passer par une analyse et une expérimentation approfondie des techniques utilisées en apprentissage profond ainsi qu'en apprentissage actif profond de manière à d'identifier les heuristiques les plus performantes.

Dernièrement, les réseaux de neurones se sont avérés très efficaces pour la classification de textes, notamment en utilisant des classifieurs basés sur une architecture Transformer. Or, si beaucoup d'approches d'apprentissage actif profond ont été évaluées pour de la classification d'images, à notre connaissance, il

n'existe que peu d'études portant sur le texte et sur ce type de classifieur.

Pour cette raison, nous allons évaluer dans ce chapitre les méthodes de réseaux profonds combinées à des approches par apprentissage actif dans le but de généraliser à de gros volumes de données les connaissances acquises sur un petit échantillon.

2 État de l'art

La classification supervisée muti-classes consiste à apprendre un modèle à partir d'un ensemble de données préalablement annotées permettant d'associer une étiquette à un exemple, puis à partir de ce modèle, de prédire une étiquette pour un nouvel exemple donné. On peut distinguer trois types d'apprentissage : l'apprentissage supervisé où l'ensemble des données d'apprentissage est entièrement étiqueté, l'apprentissage non-supervisé où l'ensemble des données n'est pas étiqueté, et l'apprentissage semi-supervisé où l'ensemble des données est partiellement étiqueté. L'étiquetage est généralement réalisé par des humains, ce qui représente une tâche fastidieuse et coûteuse, parfois impossible dans le cas de jeux de données très volumineux. Nous nous intéressons ici au processus d'apprentissage actif dont l'objectif est d'optimiser l'intervention humaine pendant la phase d'étiquetage des données. En active-learning, l'annotateur humain sera appelé l'oracle. Le processus d'active-learning consistera à poser les questions les plus pertinentes à l'oracle pour améliorer les performances du modèle, avec l'idée sous-jacente que ses réponses permettront un apprentissage rapide, précis et peu coûteux en ressource.

Il existe de nombreux classifieurs utilisés avec succès sur des données textuelles comme les classifieurs SVM, les arbres de décision, etc. Les architectures de type LSTM et CNN se sont révélées peu efficaces sur les textes. Nous avons montré l'efficience des architectures Transformer pour ce type de tâche au chapitre III.

L'apprentissage actif de modèles supervisés permet de sélectionner, durant la phase d'entraînement, les échantillons à étiqueter au lieu de les prendre au hasard dans les données encore non étiquetées. L'objectif est alors, pour atteindre les mêmes performances d'un modèle, de réduire le nombre d'exemples choisis. Cette sélection se conduit sous différentes stratégies décrites par Settles [2009], dont les plus utilisées sont les suivantes :

— Apprentissage passif (*Baseline*) : le lot d'exemples présenté à l'oracle pour être étiqueté est choisi de manière aléatoire dans l'ensemble non étiqueté ;

— Apprentissage actif par échantillonnage incertain (*Uncertainty sampling*) : le lot d'exemples est choisi en fonction de la règle de décision du classifieur

dans l'intention de repérer ceux pour lesquels le modèle est le plus incertain en se basant sur une fonction de coût comme l'entropie ;

— Apprentissage actif de type bayésien (*Deep bayesian*) : Gal et al. [2017] ont proposé une amélioration de la mesure d'incertitude précédente en utilisant la technique de *Monte Carlo dropout*. L'entropie a été de nouveau utilisée comme fonction de coût ;

— Apprentissage actif de type variation du gradient (*Expected gradient*) : Bouneffouf [2014] choisissent le lot d'exemples parmi ceux qui modifient le plus les poids du réseaux, c'est-à-dire ceux ayant la plus grande norme de gradient ;

— Core-Set : Sener and Savarese [2018] proposent à l'oracle les échantillons se répartissant le plus homogènement possible dans l'espace de représentation choisi. Pour cela, ils résolvent un problème de positionnement de k centres (*k-center problem*) en utilisant l'algorithme *Farthest-first traversal*.

— DAL (Discriminative Active Learning) : Gissin and Shalev-Shwartz [2019] modélisent la tâche d'apprentissage actif comme une classification binaire consistant à séparer les échantillons déjà proposés à l'oracle en opposition avec ceux restant à déterminer. Le classifieur binaire repose sur une architecture neuronale.

La littérature sur l'apprentissage actif appliqué à la classification de textes est importante mais porte essentiellement sur des algorithmes de classement statistique. Comme évoqué précédemment, les classifieurs de textes les plus performants sont actuellement de type « réseaux de neurones » . Aussi, il convient d'évaluer la combinaison d'une architecture de type apprentissage profond et du processus d'apprentissage actif. Cette combinaison a déjà été explorée dans le cas de la classification d'images [Wang and Ye, 2013; Sener and Savarese, 2018] ou de contenus multimédias [Budnik, 2017]. Or, peu de propositions ont été faites dans le cas de données textuelles. Nous pouvons citer les travaux de Shen et al. [2017] qui ont travaillé sur la reconnaissance d'entités nommées qui est une tâche différente de celle que nous voulons réaliser. Bang et al. [2018] ont proposé une approche basée sur un réseau RNN qui ne nécessite pas d'extraire des caractéristiques particulières mais utilise son état interne pour traiter des séquences d'entrées. Zhang et al. [2017] ont utilisé un réseau CNN selon une stratégie de type gradient attendu appliquée à la couche de description continue des mots. Siddhant and Lipton [2018] ont réalisé une large étude empirique sur l'application de l'apprentissage actif profond pour de multiples tâches, qui montre que, dans la plupart des contextes, l'apprentissage actif de type bayesien surpasse les autres approches. Nous ne retrouvons pas, dans les articles cités précédemment, d'étude combinant les derniers algorithmes d'apprentissage

actif profond avec des classifieurs neuronaux de type Transformer, comme nous nous proposons de le faire dans ces travaux. Aussi il est important aujourd'hui d'appliquer et de comparer ces techniques modernes d'active-learning avec des classifieurs neuronaux comme BERT ou XLNet sur de la classification de textes.

Pour la méthode *Uncertainty Sampling* le système d'apprentissage évalue les échantillons de l'ensemble de données d'apprentissage. Il fait un tri entre les échantillons qu'il estime les plus sûrs et les échantillons qu'il estime plus ambigus. Pour estimer le niveau d'ambiguïté, le système utilise une fonction de coût sur le vecteur de sortie. On trouve de multiples fonctions de coût de ce type dans la littérature, pour notre étude nous retiendrons l'entropie qui donne d'après nos expérimentations les meilleurs résultats. Après cette phase de tri le système demande à l'oracle une étiquette uniquement sur les échantillons incertains. L'hypothèse étant qu'il n'est pas efficace de faire appel à lui quand la prédiction sur ces échantillons semble sûre [Lewis and Catlett, 1994].

Dans ce chapitre, nous nous intéresserons donc à la mise en place d'un processus d'apprentissage actif profond, spécifique pour la classification multi-classes de données textuelles. Nous évaluerons trois stratégies d'apprentissage actif qui se sont montrées efficaces pour la classification d'images à savoir « l'entropie » , les réseaux bayésiens, la sélection d'échantillon discriminante d'apprentissage actif. Nous comparerons l'apport de ce processus d'apprentissage actif profond pour les deux architectures désormais de référence pour la classification de textes : BERT et XLNet. Nous utiliserons six corpus de textes aux particularités différentes.

3 Méthode

Les méthodes d'apprentissage actif profond fonctionnent globalement très bien pour certains types de données comme les images [Gal et al., 2017]. Il reste à adapter ces méthodes à l'analyse des textes et actuellement peu d'applications existent [Siddhant and Lipton, 2018].

3.1 Description des textes

Préalablement à cette étude, nous avons comparé sur les jeux de données de l'article les approches précédemment citées. BERT et XLNet se sont avérées les plus efficaces.

3.2 Apprentissage actif par lot

Nous considérons $\mathscr{L}$ un ensemble de données étiquetées et $\mathscr{U}$ un ensemble de données non étiquetées. Il est classique en apprentissage actif de procéder

au questionnement de l'oracle question après question. Le modèle est entraîné sur le jeu de données étiquetées. Puis, un unique nouvel exemple est proposé à l'oracle pour étiquetage. Avec un classifieur de type réseaux de neurones, ce fonctionnement n'est pas exploitable car il n'est pas pertinent d'entraîner le réseau sur un seul texte à la fois. C'est pour cette raison que nous procédons par lot. Cela revient à demander à l'oracle d'étiqueter un lot d'exemples et non plus un unique exemple. On recherche donc un ensemble d'exemples à étiqueter dans $\mathscr{U}$ qui va maximiser les performances du classifieur. Cette sélection peut se faire selon différentes approches. Soit x^* l'ensemble des instances les plus informatives selon $\varphi(x_i,\theta)$ avec φ la fonction utilisée pour évaluer les instances x_i de $\mathscr{U}$ selon l'ensemble des paramètres du modèle θ.

$$x^* = \underset{x_i \in \mathscr{U}}{\operatorname{argmax}} \varphi(x_i,\theta)$$

Le processus d'apprentissage actif peut se présenter d'après [Yang et al., 2009] sous forme d'un algorithme (voir Algorithme 1).

Algorithm 1 Multi-classe Active Learning

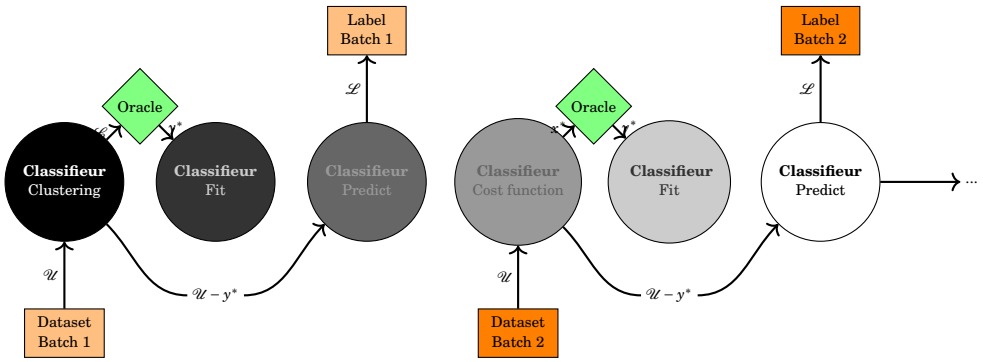
Require:

Number of batch B
Labeled set $\mathscr{L} = \emptyset$
Unlabeled set $\mathscr{U}$
Clustered set $\mathscr{C}$
Test set $\mathscr{T}$
Unlabeled subset $\mathscr{U}_t$
Number of classes C
Number of selected examples per iteration S

Acquistion $\mathscr{U}_1$
Select a set of S examples in cluster by classes C
Train classifier
Update the training set $\mathscr{L} \leftarrow \mathscr{L} \cup \mathscr{C}$
for b = 2 to B **do**
  Acquistion $\mathscr{U}_b$
  for each instance x in $\mathscr{U}_b$ **do**
    Predict its label vector using the classifier prediction
    Compute the *score* of the label vector with the cost function
  end for
  Sort *score* in decreasing order for all x in $\mathscr{U}_t$
  Select a set of x^* examples with the largest scores
  Query the oracle on $x^* \rightarrow$ obtaining y^*
  Train classifier based on training data y^*
  Update the training set $\mathscr{L} \leftarrow \mathscr{L} \cup y^*$
  Remove y^* from $\mathscr{U}$
end for
Classifier predicts the labels on $\mathscr{T}$
Evaluate the test set on $\mathscr{T}$

On peut également représenter le processus d'apprentissage actif avec une phase de démarrage par clustering comme représenté sur la figure IV.1.

FIGURE IV.1 – Représentation des étapes du processus d'apprentissage actif



3.3 Quatre stratégies d'apprentissage actif

Nous allons comparer dans ce travail quatre stratégies d'apprentissage actif profond.

La première stratégie appelée **baseline** (Apprentissage passif) consiste à proposer à l'oracle une sélection aléatoire des échantillons.

La deuxième stratégie, appelée **Uncertainty Sampling** (Apprentissage actif par échantillonnage incertain), est connue depuis longtemps [Lewis and Catlett, 1994]. Dans la littérature, plusieurs fonctions de coût ont été proposées pour le choix des exemples. Avec la mesure Least confidence [Culotta and McCallum, 2005], les échantillons les moins sûrs sont choisis pour être annotés. Pour évaluer la confiance, ces auteurs utilisent la probabilité de la classe la plus sûre de l'échantillon pour le modèle. Avec la mesure Smallest-margin [Scheffer et al., 2001], la sélection des échantillons se fait par la marge minimale entre les probabilités des classes de chaque échantillon. Avec l'Entropie [Shannon, 2001], le tri des échantillons s'opère à partir de l'estimation de la variabilité des probabilités des étiquettes. Généralement, c'est l'entropie qui est choisie car elle donne de bons résultats par son estimation qui agrège la totalité des probabilités de chaque exemple pour chaque étiquette.

Équation : Sélection par l'entropie

$$x^* = \underset{x_i \in \mathscr{U}}{\operatorname{argmax}} \Big[-\sum_c P(y_i = c|x_i;\theta).log(P(y_i = c|x_i;\theta)) \Big] \tag{IV.1}$$

Où c appartient aux classes, θ l'état du réseau.

La troisième stratégie, appelée **Deep Bayesian** (Apprentissage actif bayésien profond), a été proposée pour une tâche de reconnaissance d'entités nommées [Siddhant and Lipton, 2018]. Elle revient à sélectionner les échantillons qui maximisent le gain informationnel du modèle pour une succession de prédictions associées à différents abandons de neurone.

Équation : Sélection Bayésienne

$$x^* = \underset{x_i \in \mathscr{U}}{\operatorname{argmax}} \Big[-\sum_c (\frac{1}{T}\sum_t \widehat{p}_c^t) log(\frac{1}{T}\sum_t \widehat{p}_c^t) + \frac{1}{T}\sum_{c,t} \widehat{p}_c^t log \widehat{p}_c^t \Big] \tag{IV.2}$$

Où T représente la norme de l'ensemble des projections et $\widehat{p}_c^t = P(y_i = c|x_i;\omega^t)$ la probabilité que l'entrée x_i soit de la classe c pour un abandon de neurone ω^t.

La dernière stratégie, appelée **DAL**, a été appliquée avec succès pour la classification d'images avec un classifieur de type CNN Gissin and Shalev-Shwartz [2019]. L'approche consiste à sélectionner de façon la plus différenciée qui soit les échantillons proposés à l'oracle de ceux préalablement étiquetés.

4 Expérimentations

4.1 Jeux de données

Pour réaliser notre étude, nous avons choisi les six jeux de données présentés au chapitre I. Le tableau I.1 présente des statistiques clés sur ces six jeux de données. Les jeux de données que nous utilisons sont déséquilibrés en classe par nombre d'échantillons à l'exception du jeu SST-5 qui lui est équilibré en classe.

4.2 Pré-traitements des données

Pour chaque jeu de données, nous avons appliqué les pré-traitements suivants :

— suppression des ponctuations ;

— suppression des caractères spéciaux ;

— changement des majuscules en minuscules ;

— suppression des stop words ;

— lemmatisation consistant à ne conserver que le radical des mots, pour regrouper sous le même radical tous les mots d'une même famille.

Chaque texte peut être associé à zéro, une ou plusieurs classes selon le jeu de données. Ces classes sont utilisées comme sorties de la prédiction.

4.3 Protocole d'évaluation

Nous procédons comme suit pour chaque jeu de données. Dans un premier temps, nous extrayons 6 000 textes en respectant la pondération des classes que nous séparons en deux ensembles de textes toujours stratifiés respectivement de 25% pour l'ensemble d'entraînement et 75% pour l'ensemble de test. Le premier ensemble sert de jeu d'entraînement tandis que le deuxième ensemble sert de jeu de test. En ce qui concerne la phase d'apprentissage, le jeu d'entraînement est découpé en cinq sous-ensembles. Pour chaque sous-ensemble, une stratégie d'apprentissage actif est utilisée pour évaluer et ordonner les échantillons dans la perspective de questionner l'oracle pour une proportion de questions fixée. La fonction utilisée dépend de la stratégie d'apprentissage actif évaluée. À la suite de l'étiquetage réalisé par l'oracle, le classifieur est entraîné puis est utilisé pour étiqueter le reste du sous-ensemble sur l'ensemble de test. Pour estimer la qualité de l'apprentissage, nous utilisons la métrique d'exactitude, qui se calcule comme le rapport du nombre d'étiquettes correctement attribuées par le classifieur sur le nombre total d'étiquettes. Nous réitérons ce processus quatre fois par validation croisée et moyennons la valeur d'exactitude.

Afin de réaliser notre étude, nous disposons de plusieurs jeux de données textuels. Chacun d'eux est constitué d'un corpus de textes et chaque texte est associé à un ensemble dont nous connaissons les significations.

4.4 Hyper-paramétrage et entraînement

Dans notre étude, nous avons implémenté le système avec cinq lots de données. Ainsi, si la base contient 1 000 textes, nous pouvons interroger l'oracle sur 100 textes pour un ratio de questionnement de 0.1 soit 10%. Nous choisissons cinq paquets de vingt textes à questionner pour l'oracle parmi les 1 000. L'ensemble des hyper-paramètres du réseau est fixé pour l'ensemble des jeux de données : un taux d'apprentissage de 0.00003 avec mise à jour, époques de 20 avec procédure de sauvegarde du réseau et arrêt rapide, abandon de neurone de 0.3. Tous nos réseaux de neurones sont entraînés avec un processus de back-propagation utilisant une fonction d'erreur (*loss*) de type entropie croisée. Les optimiseurs de calcul du gradient sont de type *Adam pondéré*.

Lors de nos expérimentations préliminaires, nous avons optimisé les résultats de la base-line par l'utilisation d'une fonction d'early stopping en back-end avec un choix de départ du nombre d'époques relativement important (20).

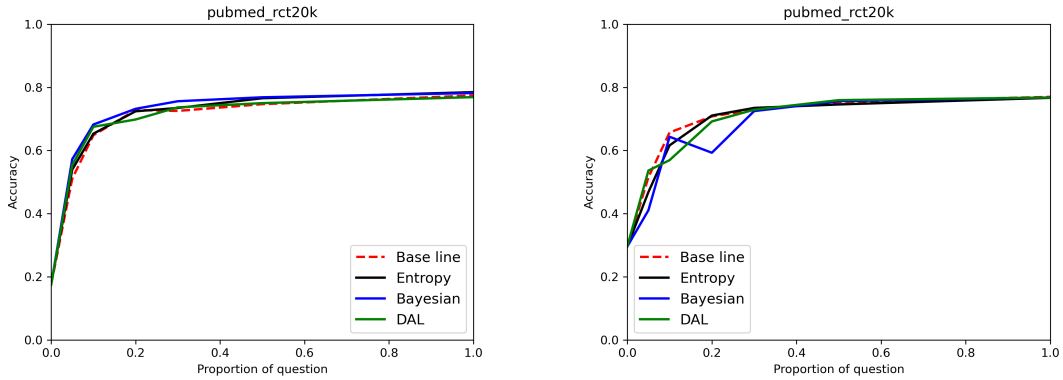


FIGURE IV.2 – L'abscisse représente la proportion des questions posées à l'oracle au regard du nombre de questions possibles et l'ordonnée, l'exactitude du classifieur. Le graphique de la première colonne correspond à l'architecture BERT tandis que la deuxième colonne correspond à l'architecture XLNet (jeu de données PubMed RCT).

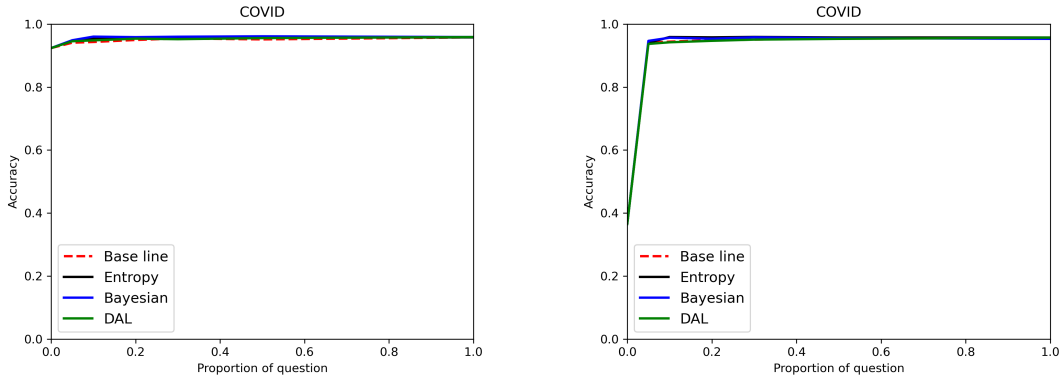


FIGURE IV.3 – L'abscisse représente la proportion des questions posées à l'oracle au regard du nombre de questions possibles et l'ordonnée, l'exactitude du classifieur. Le graphique de la première colonne correspond à l'architecture BERT tandis que la deuxième colonne correspond à l'architecture XLNet (jeu de données COVID 19).

5 Résultats des expérimentations

Dans cette section, nous reportons les résultats de nos expérimentations dans les figures IV.2, IV.3, IV.4, IV.5, IV.6 et IV.7. Dans les différents graphes, nous représentons en abscisse la proportion des questions posées à l'oracle au regard du nombre de questions possibles, et en ordonnée, nous représentons

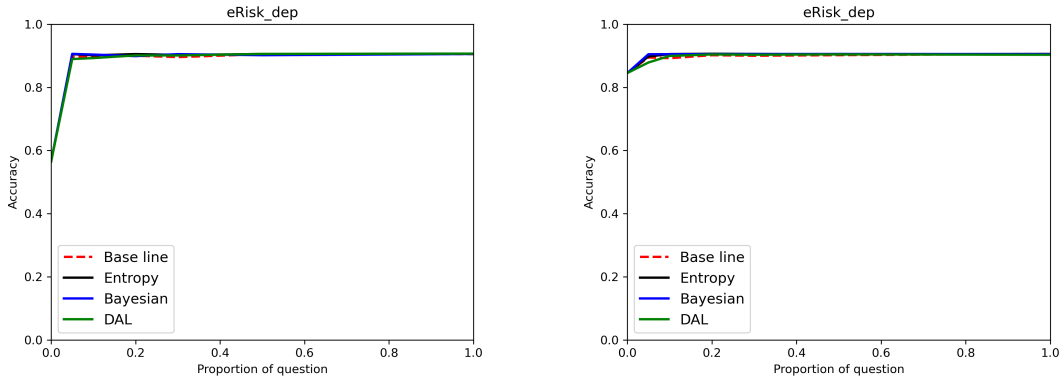


FIGURE IV.4 – L'abscisse représente la proportion des questions posées à l'oracle au regard du nombre de questions possibles et l'ordonnée, l'exactitude du classifieur. Le graphique de la première colonne correspond à l'architecture BERT tandis que la deuxième colonne correspond à l'architecture XLNet (jeu de données eRisk dépression).

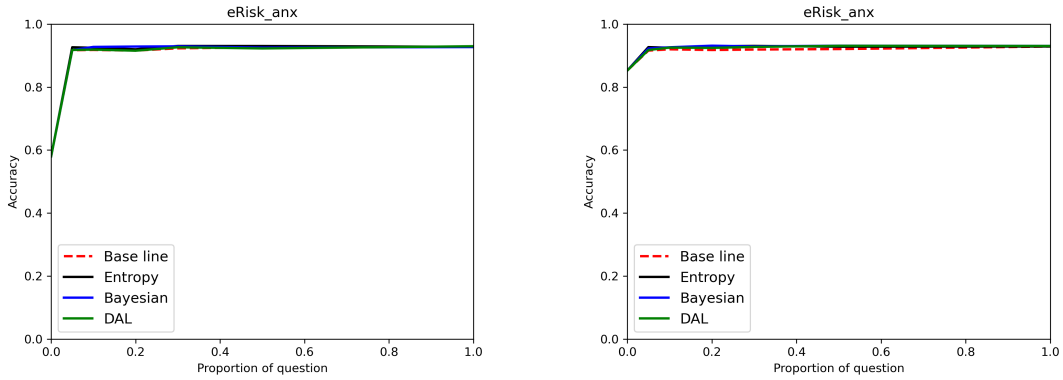


FIGURE IV.5 – L'abscisse représente la proportion des questions posées à l'oracle au regard du nombre de questions possibles et l'ordonnée, l'exactitude du classifieur. Le graphique de la première colonne correspond à l'architecture BERT tandis que la deuxième colonne correspond à l'architecture XLNet (jeu de données eRisk anorexie).

l'exactitude du classifieur.

Comme nous pouvions nous y attendre, nous constatons pour les deux architectures, une amélioration des résultats avec l'augmentation du nombre d'échantillons fournis à l'oracle. Si l'apprentissage progresse fortement pour des taux de questionnement compris entre 0% et 20% dans la quasi-totalité des situations, un palier est observé au-delà. Cela tendrait à indiquer que le classifieur ne tire

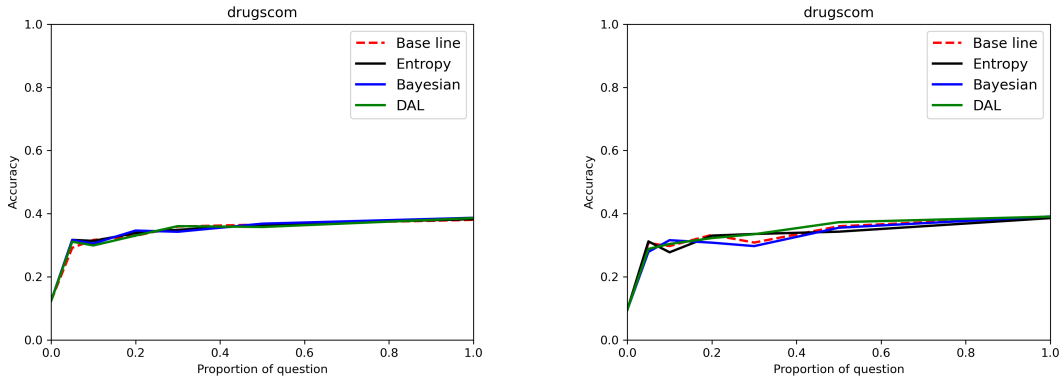


FIGURE IV.6 – L'abscisse représente la proportion des questions posées à l'oracle au regard du nombre de questions possibles et l'ordonnée, l'exactitude du classifieur. Le graphique de la première colonne correspond à l'architecture BERT tandis que la deuxième colonne correspond à l'architecture XLNet (jeu de données Drugs.com).

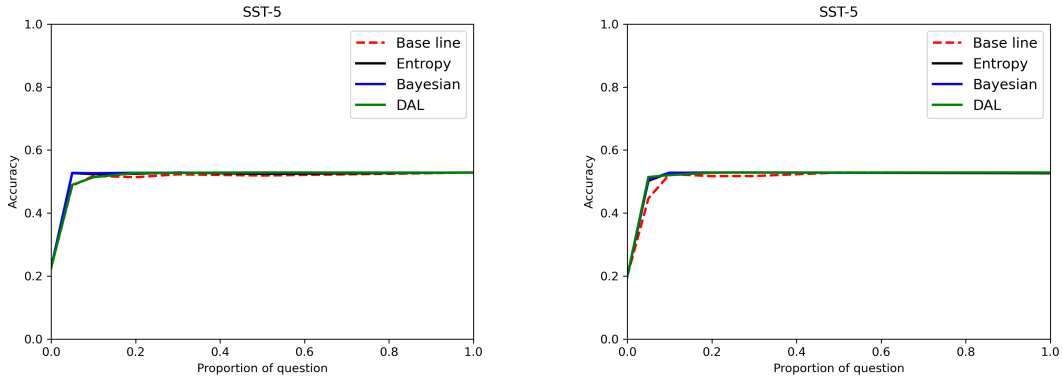


FIGURE IV.7 – L'abscisse représente la proportion des questions posées à l'oracle au regard du nombre de questions possibles et l'ordonnée, l'exactitude du classifieur. Le graphique de la première colonne correspond à l'architecture BERT tandis que la deuxième colonne correspond à l'architecture XLNet (jeu de données SST).

pas parti des exemples supplémentaires de manière linéaire. L'effort d'étiquetage par un humain devrait alors se concentrer au niveau du coude sur la courbe.

Pour le jeu de données COVID, le classifieur BERT surperforme la baseline vis à vis des stratégies d'apprentissage actif et ce pour l'ensemble des proportions de questions posées à l'oracle ce qui n'est pas le cas pour XLNet. Pour les jeux de données eRisk dépression, eRisk anorexie et drugs.com, les évolutions

graphiques sont très proches pour les deux classifieurs étudiés et il n'est pas possible de dégager une stratégie supérieure aux autres. Concernant le jeu de données SST-5, la baseline est inférieure aux stratégies d'apprentissage actif, même si les réseaux bayésiens sont très légèrement supérieurs à l'entropie et à DAL. L'écart est très faible.

Finalement, les expérimentations montrent bien une légère amélioration des stratégies d'apprentissage actif, toute fois, cette amélioration n'est que très peu visible sur les graphique et ne se retrouve pas forcément sur l'ensemble des jeux de données.

6 Conclusion

Dans ce chapitre, nous avons présenté une étude comparative des différentes stratégies d'apprentissage actif combinées à deux architectures de réseaux de neurones récentes BERT et XLNet, pour une tâche spécifique de classification de textes. Nous avons montré d'une part qu'avec les réseaux de type BERT et XLNet, les résultats étaient très proches quel que soit la stratégie d'apprentissage actif mise en œuvre. D'autre part, avec les réseaux de type BERT ou XLNet, une stratégie d'apprentissage actif par les réseaux bayésiens pourrait être la plus efficace. Bien que ces travaux restent préliminaires, ils nous permettent de suggérer l'utilisation de la combinaison de réseaux de type BERT ou XLNet et d'apprentissage actif dans le cas des petits jeux de données étiquetés.

En perspective, nous nous proposons d'utiliser d'autres architectures de type Transformer telles que BART ou T5 . De plus, nous nous interrogeons également sur l'impact de la description des textes en entrée de ce processus d'apprentissage actif profond et nous envisageons de poursuivre cette étude comparative avec des descriptions continues des mots comme GPT2 et GPT3. Toujours dans l'objectif de pouvoir traiter de petits jeux de données étiquetés, dans le chapitre suivant nous explorons d'autres approches telles que l'augmentation des données [Belohlávek et al., 2018; Abulaish and Sah, 2019] qui consiste à générer de nouvelles données d'entraînement à partir des données existantes.

V

Augmentation de données

Plan du chapitre

L'information médicale est présente dans différentes sources textuelles comme les dossiers médicaux informatisés, la littérature biomédicale, les médias sociaux, etc. Exploiter l'ensemble de ces sources pour en extraire de l'information utile représente un véritable défi. Dans ce contexte, la classification de textes est une tâche importante. Récemment, les classifieurs profonds ont montré leur capacité à obtenir de très bons résultats mais leurs résultats dépendent généralement de la quantité de données utilisée pour l'entraînement.

Dans ce chapitre, nous proposons une nouvelle approche pour augmenter les données textuelles. Nous allons comparer cette approche pour six jeux de données réels avec les principales approches de la littérature. Notre proposition sur-performe dans toutes les configurations que nous avons pu mettre en œuvre.

Le chapitre comporte cinq sections proposant la description complète de notre proposition à savoir une introduction dans la section 1, un état de l'art dans la section 2, la description de notre proposition dans la section 3, les expérimentations dans la section 4, les résultats des expérimentations dans la section 5 ainsi q'une conclusion dans la section 6.

1 Introduction

Dans ce chapitre, nous nous intéressons à l'augmentation de données qui peut s'avérer efficace sur de petits jeux de données. L'augmentation de données exploite des données en quantité limitée et transforme les échantillons existants pour en créer de nouveaux. Un challenge important consiste alors à générer de nouvelles données qui conservent une même étiquette. Plus précisément, il s'agit d'injecter de la connaissance en prenant en compte les propriétés invariantes des données par rapport à certaines transformations. Les données augmentées peuvent ainsi couvrir un espace d'entrée inexploré, éviter le sur-apprentissage et améliorer la généralisation du modèle.

Cette technique s'est avérée très efficace pour des tâches de classification d'images notamment lorsque la base de données d'entraînement est limitée. Par exemple, en reconnaissance d'images [He et al., 2015], il est bien connu que les changements mineurs dus à l'échelle, au recadrage, à la déformation, à la rotation, etc. ne modifient pas les étiquettes des données car ces changements sont susceptibles de se produire dans des observations du monde réel. Cependant, les transformations qui préservent les étiquettes pour les données textuelles ne sont pas aussi évidentes et intuitives.

Dans ce chapitre, nous présentons une nouvelle technique d'augmentation de données appliquées aux textes que nous appellerons DAIA (**D**ata **A**ugmentation and **I**nference **A**ugmentation). Cette méthode est simple à mettre en place car

elle ne nécessite pas de ressources sémantiques, ni de longues phases d'entraînements. Nous évaluerons DAIA pour divers jeux de données de santé de natures différentes et nous montrerons une amélioration par rapport à l'état de l'art, particulièrement sur les petites bases de données. Le code est public et accessible sur le site web github, nous l'avons réalisé à l'aide du module python Mantéïa [1].

2 État de l'art

L'augmentation de données a été utilisée avec succès, dans le domaine de l'analyse d'images [Shorten and Khoshgoftaar, 2019]. Par exemple, [Perez and Wang, 2017] ont comparé plusieurs techniques simples, telles que le recadrage, la rotation et le retournement d'images avec des techniques plus évoluées comme les GAN (Generative Adversarial Network) pour générer des images de différents styles ou encore des approches d'augmentation par réseau neuronal apprenant les augmentations qui améliorent le plus un classifieur. Si beaucoup de solutions existent pour l'analyse des images, les méthodes d'augmentation ont été beaucoup moins étudiées dans le domaine de l'analyse des textes. On distingue quatre approches principales que nous allons décrire ci-dessous :

— Des approches utilisant des ressources sémantiques ont tout d'abord été proposées. Par exemple, Zhang and LeCun [2015] ont utilisé un thésaurus pour remplacer les mots par leurs synonymes, afin de créer un jeu de données augmenté pour une tâche de classification de textes. Cette augmentation s'est avérée peu efficace et dans certains cas, a même diminué les performances. Les auteurs expliquent que lorsque de grandes quantités de données réelles sont disponibles, les modèles généralisent proprement et ne s'améliorent pas avec l'augmentation des données.

— Des approches inspirées des distorsions que l'on peut rajouter dans les images, ont été également appliquées aux textes. Pour une tâche de classification, Wei and Zou [2019], dans la méthode EDA, augmentent le nombre d'échantillons en supprimant, permutant un mot ou en le remplaçant par un synonyme. Les expérimentations sont faites avec des classifieurs de type CNN et RNN et montrent de réelles améliorations sur les petits jeux de données. Par exemple, l'exactitude progresse de 3% en moyenne avec le classifieur CNN pour 500 textes étudiés. Des approches se sont focalisés sur le choix des mots à modifier. Pour Kobayashi [2018], le choix des mots se fait par l'analyse du contexte du mot à échanger. Le contexte est défini à partir de l'entraînement d'un réseau de neurones de type LSTM. Cette

1. https://github.com/ym001/Manteia/blob/master/notebook/\notebook_Manteia_classification_augmentation_run_in_colab.ipynb

approche a produit une amélioration de 0.5% de l'exactitude sur cinq jeux de données. Dans la méthode UDA, Xie et al. [2019] remplacent les mots à faible contenu informationnel, repérés avec un TF-IDF faible, par leur synonyme tout en gardant ceux qui ont des valeurs de TF-IDF élevées représentant les mots clés. Cette heuristique a été testée sur six corpus de textes, ce qui a permis de réduire l'erreur de classification de 0.3% à 3%.

— Des approches génératives ont également été étudiées. Dans le domaine spécifique de la réponse visuelle aux questions (Visual Question Answering VQA), qui consiste à partir d'une image et d'une question sur cette image, à prédire la réponse à la question. Kafle et al. [2017] ont aussi mis en place deux approches pour générer de nouvelles questions. Pour cela, ils utilisent des annotations sémantiques existantes combinées avec une approche générative utilisant des réseaux de neurones récurrents. Leurs deux méthodes améliorent de 1 à 2% l'exactitude en augmentant la variété et le nombre de questions. On retrouve également les travaux de Gupta [2019] qui entraîne des modèles GAN sur des ensembles de données de faibles tailles, puis les utilise à des fins d'augmentation des données pour améliorer la généralisation d'un classifieur de sentiments. Les résultats améliorent l'exactitude de l'ordre de 1% sur les deux jeux de données étudiés.

— Une dernière approche se base sur l'augmentation de données textuelles à l'aide de la traduction inversée (back-translation) [Sennrich et al., 2015; Edunov et al., 2018]. Il s'agit de traduire un exemple dans un langage puis de retraduire la traduction obtenue dans le langage initial. Yu et al. [2018] expliquent ainsi que la traduction inversée génère des paraphrases en préservant la sémantique des phrases d'origine. Sur le jeu de données SQAD (Stanford Question Answering Dataset), ils obtiennent une amélioration jusqu'à 3% de l'exactitude. Cependant, Shleifer [2019] a montré que la technique de traduction inversée n'apportait que peu d'améliorations avec les classifieurs modernes comme UMLfit.

Dans ce chapitre, nous allons nous focaliser ici sur une approche de distorsion, simple à mettre en place, ne nécessitant pas de ressource comme dans les approches sémantiques, ni de grandes quantités de calculs comme les approches génératives ou encore l'accès à des ressources externes comme les approches de traduction inversée. Une limite des approches de distorsion que l'on retrouve dans la littérature est qu'elles ne préservent pas l'ordre des mots dans les phrases. Les exemples alors générés s'éloignent de ceux que l'on pourrait retrouver dans le monde réel. Dans l'approche DAIA proposée, nous allons décrire trois approches pour découper les phrases du jeu d'apprentissage en plusieurs séquences qui seront utilisées pour l'augmentation. La même approche

sera utilisée dans la phase de test sur les exemples à classifier en appliquant une technique de soft voting.

Nous montrerons dans les expérimentations que cette approche améliore les résultats d'une classification de textes quel que soit le type de classifieurs profond utilisé comme BERT [Devlin et al., 2019], RoBERTa [Liu et al., 2019], AlBERT [Lan et al., 2019], DistilBERT [Sanh et al., 2019] ou encore ScienceBERT [Beltagy et al., 2019]. Nous comparerons également les expérimentations de notre proposition DAIA avec les approches par distorsion UDA [Xie et al., 2019] et EDA [Wei and Zou, 2019], l'approche générative TextGen [Gupta, 2019] ou encore la traduction inversée [Shleifer, 2019].

3 Méthode : DAIA

Notre proposition DAIA s'articule en deux parties : la première porte sur la phase d'entraînement (DA Data Augmentation) et la deuxième sur la phase de test (IA Inference Augmentation).

Data augmentation : Dans la phase d'entraînement, nous augmentons la quantité de données en découpant le texte initial de chaque échantillon. Nous cherchons à produire de nouveaux échantillons sans modifier la relation d'ordre existant dans la succession des mots de chaque phrase de manière à ne pas diminuer la qualité d'apprentissage de description des représentations continues de mots. Dans la phase d'entraînement, nous avons appliqué trois approches pour découper la phrase initiale en n séquences de mots qui seront associées chacune à la même étiquette que la phrase initiale. Ces trois approches sont présentées dans la suite de ce paragraphe et un exemple est décrit dans la figure V.1.

— Découpe 1 symétrique : Nous découpons de manière symétrique le texte en supprimant une portion de $x\%$ du texte aux deux extrémités pour générer une nouvelle séquence de texte. Pour chaque texte initial, nous obtenons ainsi un texte supplémentaire.

— Découpe 2 par fenêtre glissante : Nous appliquons un découpage par fenêtre glissante de taille l qui se décale de m mots jusqu'à la fin du texte initial. Le nombre de séquences générées dépend de la longueur de la phrase initiale.

— Découpe 3 en parties égales : Le découpage se fait en i parties égales. Nous obtenons ainsi i nouveaux documents plus le document original.

À l'issue, de la phase d'augmentation de la phase d'entraînement, à partir de chaque texte du jeu d'entraînement initial, nous avons généré un ensemble de nouvelles séquences qui sont associées à la même étiquette que le texte initial et sont utilisées en entrée du modèle d'apprentissage.

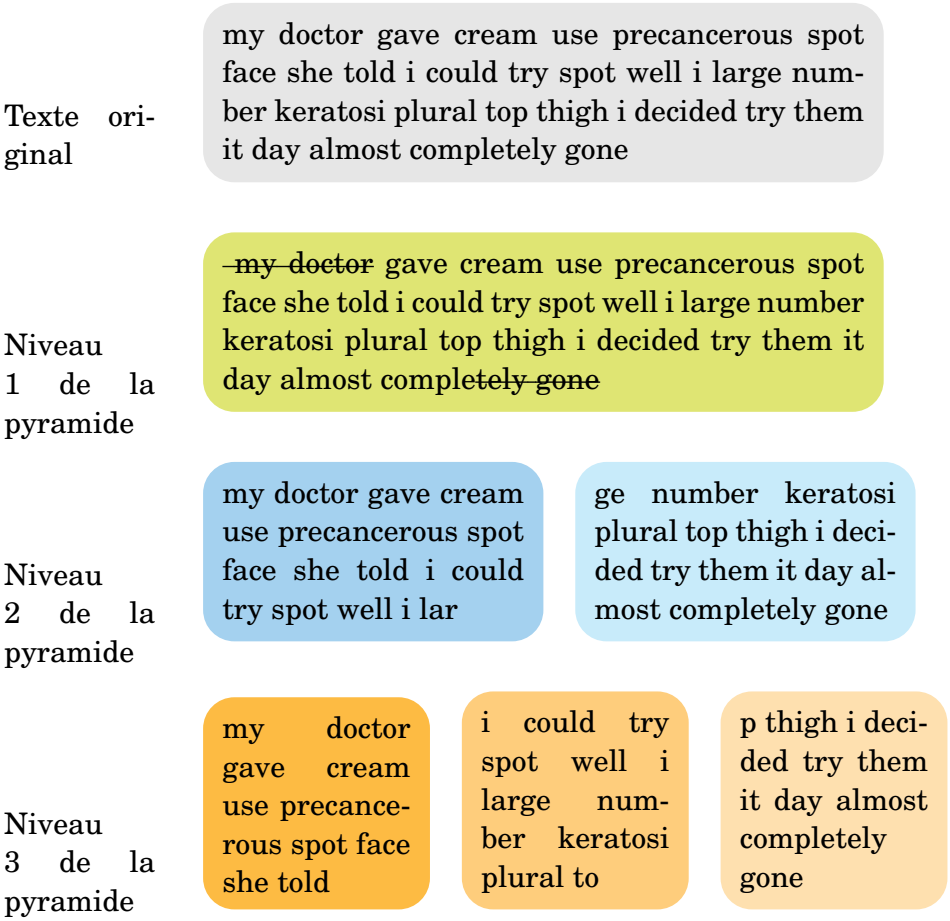


FIGURE V.1 – Description de la découpe 3 pyramidale. La figure illustre un découpage en 3 niveaux, permettant ainsi d'obtenir 6 nouveaux documents (plus le texte original).

Nous avons testé les avantages et les inconvénients de ces différents types de découpage et avons finalement proposé une découpe pyramidale qui combine les découpes 1 et 3. Cette nouvelle découpe se décline en n niveaux. L'augmentation pour le niveau 1 se fait par la découpe symétrique 1. L'augmentation pour le niveau 2 se fait en découpant le texte en deux parties égales et en rajoutant l'augmentation obtenue au niveau 1. L'augmentation pour le niveau i se fait en découpant le texte en i parties égales et en rajoutant l'augmentation de niveau $i-1$ comme illustré par la figure V.1. Pour chaque texte, nous obtenons ainsi $\frac{n\times(n+1)}{2}$ nouveaux documents étiquetés.

Inference augmentation : La phase de test consiste à prédire des étiquettes pour de nouveaux textes à partir du modèle appris pendant la phase d'apprentissage. Nous découpons le texte de l'exemple à classer en suivant le même protocole que pendant la phase d'entraînement, puis nous donnons au

classifieur l'ensemble des séquences de textes générées. Pour chacune de ces séquences, le classifieur renvoie un ensemble de prédictions qui sont agrégées pour le texte initial par soft-voting (somme des valeurs pour chaque classe prédîtes de chaque élément de l'ensemble). Ainsi, pour chaque texte à classer, nous obtenons une valeur par classe et retenons l'étiquette associée à la valeur maximale des prédictions obtenues par classe.

4 Expérimentations

4.1 Jeux de données

Nous avons utilisé pour ces expérimentations les jeux de données présentés au chapitre I page 22.

4.2 Pré-traitements des données

Pour chaque jeu de données, nous avons appliqué les pré-traitements suivants : suppression des ponctuations, des caractères spéciaux, des stop words, changement des majuscules en minuscules et lemmatisation avec le module python NLTK[2]. Chaque texte peut être associé à zéro, une ou plusieurs classes selon le jeu de données. Ces classes sont utilisées comme sortie de la prédiction.

4.3 Comparaison à d'autres approches de l'état de l'art

Nous comparons notre proposition à trois méthodes de l'état de l'art que nous décrivons en détail ci-dessous.

Pour les approches par distorsion sémantique de mot, nous avons considéré les approches EDA et UDA.

Pour EDA, nous mettons en œuvre les quatre techniques simples de data augmentation décrites par [Wei and Zou, 2019] à savoir le remplacement par des synonymes, l'insertion aléatoire, l'échange aléatoire, la suppression aléatoire. Pour cela, nous avons utilisé le dictionnaire de synonymes Wordnet de NLTK[3].

Pour UDA, dans le but de repérer les mots ayant le plus de contenu informationnel, nous procédons comme [Xie et al., 2019] qui repèrent ces mots comme étant ceux négativement corrélés à leur score TF-IDF. Pour cela, ils définissent la probabilité $min(p(C - TF\text{-}IDF(xi))/Z, 1)$, où p est un hyper-paramètre contrôlant la variation d'augmentation, C est le score maximum de TF-IDF pour les mots x_i d'un texte x et $Z = \sum_i (C - TF\text{-}IDF(xi))/|x|$. Pour nos expérimentations, nous avons choisi $p = 0.9$. Xie et al. ne modifient pas les mots clés, repérés par

2. https ://www.nltk.org/
3. https ://www.nltk.org/howto/wordnet.html

leur fréquence. Les mots repérés, qui ne sont pas des mots clés, sont remplacés par un des mots non essentiels du corpus.

Comme les générateurs de textes de l'état de l'art utilisés pour l'augmentation des données ont été dépassés en qualité sémantique par le générateur GPT2 [Gupta, 2019], nous avons utilisé ce dernier selon le protocole suivant. Pour chaque texte, un texte augmenté est construit par concaténation de deux parties. Une partie initiale, la graine, correspondant à la moitié du texte original et une deuxième partie obtenue en utilisant le générateur GPT2 ayant pris en entrée la graine.

L'augmentation de données par traduction inversée [Shleifer, 2019] consiste à produire des paraphrases conservant globalement la sémantique de la phrase initiale. Nous utilisons le service web de translation Yandex [4] pour produire ces paraphrases. Sugiyama and Yoshinaga [2019] ont monté que la qualité d'une traduction inversée s'améliore lorsque les langues sont relativement éloignées de par leur structure ; aussi nous avons choisi une traduction inversée Anglais-Japonais-Anglais identiquement à leurs travaux. Nous procédons à une première traduction des textes en Japonais puis nous retraduisons ces textes en Anglais et enfin nous les étiquetons avec le label original du texte.

4.4 Protocole d'évaluation

Nous procédons comme suit pour chaque jeu de données décrit dans la section 4.1. Nous extrayons 5 000 textes en respectant la pondération des classes. Ensuite, nous séparons en deux ensembles les données : un premier ensemble de 1 250 textes (soit 25%) utilisé pour la phase de test et un deuxième ensemble de 3 750 textes (soit 75%) pour l'apprentissage en respectant la stratification.

Pour estimer la qualité de l'apprentissage, nous utilisons la métrique d'exactitude, qui se calcule comme le rapport du nombre d'étiquettes correctement attribuées par le classifieur sur le nombre total d'étiquettes. Toutes les valeurs produites dans cette étude ont été calculées sur la moyenne d'une validation croisée à quatre plis.

Il existe un lien assez important entre la taille du jeu d'entraînement, le nombre d'époques d'entraînement et le sur- ou sous-apprentissage. Pour faire une comparaison entre les réseaux sans augmentation et ceux avec, nous avons adapté le nombre d'époques d'apprentissage comme indiqué dans le tableau V.1.

4. https ://yandex.com/

TABLE V.1 – Profil d'entraînement des réseaux de neurones.

Taille des jeux de données	50	500	5000	50 000
sans DA	128	32	8	2
avec DA	16	4	2	1

4.5 Hyper-paramétrage et entraînement

La taille des lots de données d'entraînement est fixée à 8 pour des limitations de taille de mémoire. L'ensemble des hyper-paramètres des réseaux est fixé pour l'ensemble des jeux de données. Le taux d'apprentissage est fixé à 0.00001. Tous nos réseaux de neurones sont entraînés avec un processus de propagation d'arrière en avant utilisant une fonction d'erreur (*loss*) de type entropie croisée. Les optimiseurs de calcul du gradient sont de type *Adam weight*. De plus, nous utilisons une mise à jour du taux d'apprentissage linéaire avec une amélioration du départ. Les expérimentations sont faites sur deux GPU de type GeForce RTX.

5 Résultats des expérimentations

5.1 Expérimentations préliminaires

Nous avons débuté nos expérimentations par une phase préliminaire ou nous avons évalué trois types de découpes proposées sur le jeu de données PubMed 200k RCT pour 1 000 textes. Nous avons obtenu des résultats similaires avec les autres jeux de données de l'étude. Nous considérons dans les trois tableaux suivants, la baseline comme étant le classifieur BERT sans augmentation de données.

Évaluation de la découpe 1 symétrique : Nous avons cherché à évaluer l'influence de la taille de la portion de texte supprimée aux extrémités, que nous avons fait varier comme suit 5%, 15%, 25%, 35%, 45%. Les résultats sont présentés dans la table V.2. La meilleure exactitude est obtenue pour les suppressions de textes les plus petites, ce qui laisse penser que l'augmentation fonctionne mieux lorsque les modifications du texte initial sont les moins importantes.

TABLE V.2 – Découpe 1 symétrique : étude selon la proportion de textes supprimés par échantillon aux extrémités.

Taille de la découpe(%)	5	15	25	35	45
Baseline sans DA	0.6513	0.6513	0.6513	**0.6513**	**0.6513**
DA - découpe 1 symétrique	**0.6666**	**0.6543**	**0.6546**	0.6506	0.6469

Évaluation de la découpe 2 par fenêtre glissante : Nous avons évalué pour la même taille de fenêtre, 90% du texte initial, un décalage de 1 à 5 mots. Les résultats sont présentés dans la table V.3. Les meilleurs résultats sont obtenus avec le décalage de 4 mots.

TABLE V.3 – Découpe 2 par fenêtre glissante : étude selon le pas utilisé pour faire glisser la fenêtre

Pas entre fenêtre (mot)	1	2	3	4	5
Baseline sans DA	0.6513	0.6513	0.6513	0.6513	0.6513
DA - découpe 2 fenêtre	**0.6586**	**0.651**	**0.6533**	**0.663**	**0.662**

Évaluation de la découpe 3 en partie égale : Nous avons évalué l'impact du nombre de découpes en 2, 3, 4, 5 et 6 parties. Les résultats sont présentés dans la table V.4. Nous constatons que la division par deux du document donne les meilleurs résultats.

TABLE V.4 – Découpe 3 par découpe en parties égales

Nombre de parties	2	3	4	5	6
Baseline sans DA	0.6513	0.6513	0.6513	0.6513	0.6513
DA - découpe 3 pyramidale	**0.677**	**0.659**	**0.6513**	**0.6553**	0.6283

Nous retenons de ces expérimentations préliminaires que la découpe 1 avec le plus petit pourcentage de découpe ainsi que la découpe 3 en 2 portions de textes. Dans la suite de nos expérimentations, nous utiliserons une combinaison de ces deux découpes que nous appellerons découpe pyramidale, que nous avons décrite dans la section 2.

5.2 Impact du classifieur et comparaison à l'état de l'art

Nous travaillons dans la suite sur le jeu de données drugs.com avec un échantillon de 5 000 textes. Nous avons obtenu des résultats similaires avec les autres

TABLE V.5 – Étude en fonction du classifieur et comparaison aux approches de la littérature

Classifieur	RoBERTa	BERT	XlNet	AlBERT	DistilBERT	Scibert
Baseline sans DA	0.3661	0.3637	0.3760	0.3200	0.3601	0.3392
Distorsion EDA	0.38	0.3669	0.3712	0.3226	0.3689	0.3510
Distorsion UDA	0.3811	0.3632	0.3525	0.3084	0.3660	0.3327
Génération de textes GPT2	0.3384	0.3252	0.3425	0.3122	0.3204	0.3078
Traduction inversée	0.3798	0.3462	0.3728	0.3426	0.3584	0.3361
DA - découpe symétrique	0.3769	0.3491	0.3568	0.3154	0.3359	0.3498
DA - découpe glissante	0.3494	0.3399	0.3657	0.3266	0.3527	0.3156
DA - découpe pyramidale	0.3877	0.3660	0.3762	0.334	0.3688	0.3590
DAIA	**0.3931**	**0.3755**	**0.3786**	**0.3444**	**0.3693**	**0.3625**

jeux de notre étude. La baseline correspond à l'utilisation d'un classifieur seul sans augmentation de données. Les classifieurs comparés sont les classifieurs les plus performants de la littérature : BERT [Devlin et al., 2019], RoBERTa [Liu et al., 2019], AlBERT [Lan et al., 2019], DistilBERT [Sanh et al., 2019] ou encore ScienceBERT [Beltagy et al., 2019]. La méthode DAIA est appliquée ici pour l'augmentation des données pendant la phase d'apprentissage mais également pendant la phase de test comme détaillé dans la section 3.

Les résultats sont présentés dans le tableau V.5. La découpe pyramidale surperforme les autres découpes quel que soit le classifieur. Elle est aussi supérieure aux autres approches de la littérature comme EDA et UDA ou encore la traduction inversée, pour les réseaux RoBERTa et XlNet. Combinée avec l'inférence augmentation, DAIA surperforme pour tous les classifieurs. Les réseaux les plus importants en termes de nombre de neurones sur-performent les autres et parmi eux, c'est le réseau RoBERTa qui dépasse les autres en terme d'exactitude.

5.3 Impact du nombre de classes

Dans le tableau V.6, nous étudions les performances de DAIA en fonction du nombre de catégories pour les cinq corpus de l'étude. Nous avons retenu Roberta comme classifieur avec un échantillonnage de 500 textes. On remarque une amélioration apportée par DAIA plus importante pour les jeux ayant un nombre de classes élevé comme Drugs.com (10).

TABLE V.6 – Impact du nombre de classes

Jeux	Drugs.com	SST	PubMed 200k RCT	WHO COVID-19	eR Depression	eR anoxerie
Base line	0.2846	52.6	0.6413	0.9319	0.8926	0.9079
DAIA	**0.316**	**55**	**0.6513**	**0.938**	**0.9066**	**0.9153**

5.4 Impact de la quantité de données dans le jeu d'apprentissage

Nous travaillons ici avec comme baseline le classifieur RoBERTa sans augmentation de données. Nous montrons dans la figure V.2 que DAIA apporte une amélioration pour toutes les tailles des jeux de données étiquetées Drugs.com (à gauche) et eR anorexie (à droite). En particulier, dès les jeux de données de 500 textes, on observe une amélioration de 2.7%. L'impact se réduit quand les jeux atteignent la taille des 5 000 textes.

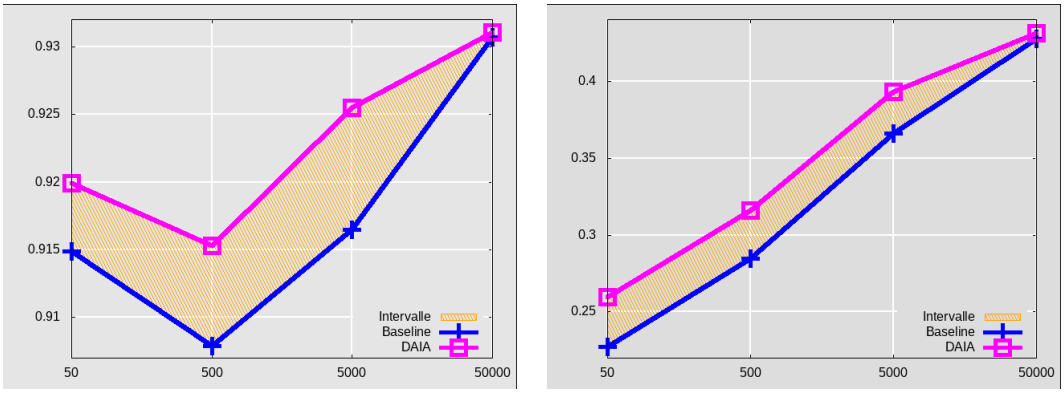


FIGURE V.2 – Étude en fonction de la taille du corpus.

5.5 Impact de la taille de séquence retenue

Dans la figure V.3, nous étudions les variations de DAIA selon la taille des découpes obtenues avec l'approche 3 et la longueur des textes en entrée pour le jeu de données drugs.com. Une découpe trop petite n'apporte pas d'amélioration. Un maximum semble atteint pour un niveau de découpe trois associé à une longueur de séquence de 128 sous-mots. La taille du texte en entrée a finalement peu d'impact dans l'intervalle étudié avec un très léger maximum pour la valeur de 128 sous-mots. Nous conseillerons en conséquence pour les personnes voulant tester ou mettre en œuvre DAIA, un niveau trois de découpe pyramidal avec un maximum de taille de séquence de 128.

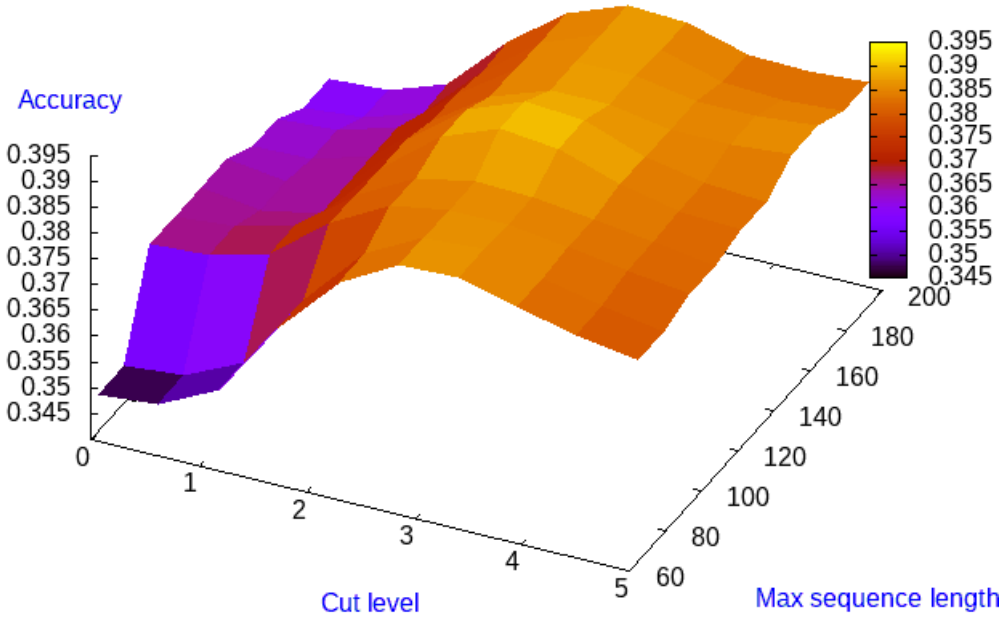


FIGURE V.3 – Étude en fonction du niveau de découpe pyramidal et de la longueur maximale de séquence en entrée du réseau.

6 Conclusion

Dans ce chapitre, nous avons proposé une nouvelle méthode pour augmenter les données textuelles, qui consiste à découper les textes en plusieurs segments pour augmenter la variété des textes d'entraînement tout en préservant la qualité de l'apprentissage des représentations continues de mots. Cette méthode, appelée Data Augmentation Inférence Augmentation (DAIA) est une approche par distorsion qui ne nécessite pas de ressource sémantique, ni de phase d'entraînement très importante, ni l'appel à des ressources extérieures. Notre approche DAIA a amélioré les performances pour six jeux de données dans le domaine de la santé, pour six différents classifieurs et a été comparée avec succès avec les principales approches de la littérature.

Les algorithmes d'augmentation de données efficients comme EDA ou UDA focalisaient leurs actions sur la modification des mots du texte à classifier. Si cette modification permet bien d'améliorer légèrement la tâche de classification, elle dégrade la construction de l'espace de description continue des mots du réseau. En effet, les réseaux de neurones depuis [Mikolov et al., 2010] utilisent principalement une fenêtre de mots pour construire leurs vecteurs de description des mots. Dans ces techniques, il est primordial de bien configurer les fenêtres de mots. Pour les réseaux comme BERT [Devlin et al., 2019], il a été introduit le "Masked language modeling" qui est un exemple d'auto-encodage consistant à prédire l'absence de 15% des mots du contexte du mot étudié. En modifiant les mots du contexte les algorithmes comme EDA ou UDA modifient le contexte des mots des phrases. Notre proposition de découpe pyramidale certes, modifie le texte mais conserve à l'intérieur de la découpe la relation d'ordre des mots et des sous-mots.

De plus, la longueur des textes numérisés en entrée des réseaux de neurones est fixe par construction alors qu'un texte écrit par une personne n'a pas de limite en longueur. Il en résulte la nécessité de limiter la taille des séquences en entrée des réseaux. Typiquement, pour BERT, cette taille des séquences est de 512. Clairement, cette limitation des classifieurs neuronaux ne permet pas de tenir compte de l'information contenue dans la partie du texte qui a du être coupée. Notre proposition, la découpe pyramidale, permet d'accéder à ce type d'information pour les textes longs. En effet, par exemple pour la découpe de niveau deux, nous utilisons le début et la fin du texte. La partie de fin du texte est alors utilisée pour entraîner le classifieur ainsi que pour émettre une prédiction sur le texte. L'utilisation de cette partie éliminée par les autres algorithmes nous semble être un élément essentiel de notre proposition.

Nous prévoyons des expérimentations en prolongement de nos travaux. L'impact du choix de la langue sur la retro-traduction doit être évalué. Une expé-

rimentation supplémentaire associée à notre approche consisterait à faire varier les découpes sur les textes, selon que l'on s'efforce de coller au plus près des textes initiaux ou au contraire que l'on se permette de générer des textes peu plausibles mais qui permettraient néanmoins d'améliorer le classifieur. Par ailleurs, il est possible de conserver le sens sans conserver l'ordre de mots. On pourrait alors imaginer de nouvelles expérimentations consistant à conserver certains niveaux de l'articulation syntaxique plutôt que l'ordre des mots dans le but de générer plus de variabilité tout en conservant la sémantique de la phrase. Nous aimerions également poursuivre ces recherches vers d'autres types de données, comme les images ou les sons [Phuong and Dat, 2013] et nous intéresser à des tâches plus complexes comme la classification multi-étiquettes, à des fins d'amélioration de classification. Enfin, il serait également très intéressant d'appliquer DAIA combinée aux heuristiques de deep active learning dans le domaine médical [Maldonado and Harabagiu, 2019] telles que discutées dans le chapitre précédent.

VI

Conclusions et perspectives

Plan du chapitre

L'information médicale est principalement entreposée sur des supports écrits. Nous nous sommes attelés dans cette thèse à améliorer l'analyse de documents de santé textuels et en particulier nous nous sommes demandés comment nous pouvions les classer efficacement de manière supervisée. Nous avons considéré deux types de jeux de données, des articles de la littérature biomédicale et des textes produits par les patients dans les médias sociaux.

1 Résumé des contributions

Dans le chapitre III, nous avons en premier lieu vérifié la pertinence des classifieurs récents sur ce type de documents. Les plus performants en terme d'évaluation par l'exactitude d'après nos expérimentations ne sont pas les classifieurs statistiques ou la méthode d'analyse peut être modélisée et bien définie comme les SVM ou les arbres de décisions. Nos expérimentations ont montré que les classifieurs à base d'architecture neuronale donnent les meilleurs résultats [Mercadier et al., 2019, 2018] en particulier l'architecture Transformers, avec des représentations de textes récentes n'impliquant pas d'ingénierie des caractéristiques. Malheureusement, il est difficile d'interpréter les résultats de ces classifieurs et de pouvoir généraliser leurs résultats, notamment lorsque l'on dispose de peu d'exemples annotés.

Dans le chapitre IV, nous nous sommes donc intéressés au cas des bases de données de textes de santé faiblement annotées. En effet, la phase nécessaire à l'apprentissage du classifieur peut s'avérer longue et fastidieuse. Si les classifieurs neuronaux reposant sur l'architecture Transformers sont bien les plus performants ceux-ci ont été imaginés par les services de recherche et développement des grandes entreprises mondiales du numérique les GAFAM. Ces entreprises gèrent des bases de données très importantes. On peut citer Facebook qui dispose d'un nombre d'inscrits dépassant le milliard [Houghton et al., 2020]. Par exemple, le développement de réseau de neurones comme BERT est orienté vers les très grosses bases de données et ainsi les résultats sur de petites bases ne sont pas aussi efficaces. Afin de remédier à ce problème, nous avons dans un deuxième temps évalué les processus d'apprentissage actif profond. La littérature nous a montré l'efficience de ces heuristiques pour réduire l'effort d'annotation nécessaire à la classification supervisée pour les classifieurs statistiques ainsi que pour les classifieurs neuronaux de type LSTM et CNN [Mercadier et al., 2020a]. Cependant, nos expérimentions ont montré une très légère amélioration avec les classifieurs de type Transformers et avec un entraînement performant cette amélioration n'est d'ailleurs pas réellement significative sur la totalité des jeux de données expérimentés.

Dans le chapitre V, nous avons recherché une amélioration directe de la clas-

sification supervisée sur de petites bases de données annotées. Pour cela nous avons mis en œuvre une méthode d'augmentation de données. L'état de l'art de cette approche nous a permis de vérifier sa pertinence sur les données de type images [Han et al., 2019] ou textes [Guo, 2020]. Les grandes entreprises commencent à s'intéresser à ce type problématique. Par exemple, le service de recherche de Google a proposé UDA [Xie et al., 2019] qui s'est avéré efficace sur des jeux de données textuels variés. Cependant, il n'existe que peu d'études sur ce sujet précisément. Nous avons donc recherché une méthode qui ne modifie pas la distribution des données initiales, qui améliore le modèle d"apprentissage et qui prenne en compte la séquentialité des données textuelles. Nous avons innové en proposant une découpe pyramidale des textes, en associant aux textes découpés une étiquette identique à celle du texte original. Nous avons amélioré notre proposition en utilisant cette découpe sur la phase d'inférence de la classification. Dans nos expérimentations, avec de nombreuses architectures de la littérature, les résultats ont dépassés en terme d'évaluation les heuristiques précédentes du domaine [Xie et al., 2019; Wei and Zou, 2019].

En résumé, la classification supervisée de textes dans le domaine de la santé est importante pour analyser les grands volumes de textes disponibles mais reste difficile du fait du peu de données préalablement annotées. Les approches proposées dans ce manuscrit ont mis en évidence deux principales observations. La première concerne les architectures à base de réseaux de neurones, notamment l'architecture Transformers, qui s'avèrent très efficaces. Or, ces approches sont fortement impactées par la taille des sources de données utilisées pour l'apprentissage. Dans ce contexte, une deuxième observation est que les approches par apprentissage actif et les approches d'augmentation de données s'avèrent très efficaces pour limiter le travail d'annotation des experts.

Les résultats présentés dans ce manuscrit sont très encourageants et peuvent être appliqués à d'autres domaines que la santé ; il existe cependant des limites et beaucoup de travail reste à faire :

— Dans le chapitre I, nous avons présenté, les jeux de données de la thèse ainsi que les tâches d'apprentissage supervisé associées, qui devraient être étendus à d'autres types de données de santé comme les documents composant les données des patients et d'autres tâches de classification spécifiques à ces textes ;

— Dans le chapitre II, nous avons décrit les principales architectures utilisées en 2020 pour l'apprentissage profond dans le traitement automatique du langage mais d'autres ont de nouveau récemment émergées ;

— Dans le chapitre III, nous avons expérimenté avec succès les approches présentées dans le chapitre II pour la classification supervisée de nos jeux de données. Cette étude reste préliminaire. Une étude plus approfondie

sur les différents types de préparation des données, paramètres des algorithmes d'apprentissage permettrait d'affiner encore l'interprétation des résultats des phases d'apprentissage et de test ;

— Dans le chapitre IV, nous avons présenté une étude comparative des différentes stratégies d'apprentissage actif combinées à deux architectures de réseaux de neurones récentes BERT et XLNet. Pour confirmer nos résultats, nous devrons intégrer des architectures Transformers plus récentes telles que BART ou T5 et nous interroger sur l'impact de la description des textes en entrée du processus d'apprentissage actif profond avec des descriptions continues des mots de type GPT2 et GPT3 ;

— Dans le chapitre V, nous avons exploré une nouvelle approche d'augmentation de données dont la principale limite est la génération arbitraire des différentes découpes qui pourrait être améliorée en faisant varier ces dernières selon que l'on s'efforce de coller au plus près des textes initiaux ou au contraire de générer des textes peu plausibles mais qui permettraient néanmoins d'améliorer le classifieur. Par ailleurs, toujours en essayant de préserver la sémantique de la phrase mais cette fois sans préserver l'ordre de mots, nous pourrions expérimenter de nouvelles approches prenant en compte certains niveaux de l'articulation syntaxique plutôt que l'ordre des mots dans le but de générer plus de variabilité ;

2 Perspectives

Les pistes de recherches que nous avons explorées dans cette thèse devront, dans l'avenir, être approfondies. En apprentissage automatique, lorsque peu de données sont disponibles, d'autres solutions que l'active learning et l'augmentation de données ont été proposées dans la littérature mais restent peu explorées sur les données de nature textuelle. Dans cette section, nous listons cinq principales orientations futures à partir de ce travail.

2.1 Annotation semi-automatique

Pour de nombreuses tâches de classification supervisées, même si les jeux de données ne sont pas étiquetés manuellement par des experts, il est possible de générer des étiquettes automatiquement [Khodak et al., 2018], basées sur des méta-données (e.g. code PMSI associé à un document de dossier patient, hashtags pour des posts Twitter...), sur des marqueurs iconiques (e.g. les emojis présents dans les messages des médias sociaux, les mots en majuscules, etc.) ou encore sur des marqueurs linguistiques que l'on peut retrouver dans le texte « Mr X a été diagnostiqué avec un ... » dans un compte rendu d'hospitalisation

ou encore « étude multi-centrique » pour les articles médicaux. Dans le cas des jeux de données utilisés dans cette thèse, la génération de ces étiquettes automatiques serait spécifique à chaque type de données et à chaque tâche de classification. Toutefois, combinées avec des approches basées sur les mécanismes d'attention [Ragheb et al., 2019], qui visent à identifier quelle partie du texte impacte le plus sur la classification, ces marqueurs pourraient être identifiés semi-automatiquement.

2.2 Auto-apprentissage

Une autre solution consiste à mettre en place des approches d'apprentissage semi-supervisées qui utilisent un ensemble de données étiquetées et non étiquetées par exemple pour le clustering de textes [Haridas and Kim, 2020] ou la segmentation sémantique. Dans ce contexte, l'auto-apprentissage (Self-training) [McClosky et al., 2006] s'avère particulièrement intéressant. Cette approche itérative et continue, exploite les prédictions les plus fiables du modèle (pseudo-étiquettes) sur des données non étiquetées pour augmenter le nombre d'exemples utilisés pendant l'entraînement. Ce processus vise à maximiser les prédictions du modèle sur des ensembles de test et de validation considérés séparément. Ces approches peuvent être judicieusement combinées avec les approches d'annotation automatique précédemment décrites. Par ailleurs, comme il est difficile dans le cas d'architectures basées sur des réseaux de neurones d'identifier les prédictions les plus fiables en se basant sur un unique modèle, des techniques récentes utilisent des ensembles de modèles. On peut citer les travaux sur la classification de sentiments par co-training [Bai et al., 2020].

2.3 Explicabilité des prédictions

Nous avons montré expérimentalement que les progrès de l'apprentissage profond ont produit une percée significative sur les scores d'exactitude de classification de texte. Cependant, on en sait peu sur le traitement de l'information interne de ces modèles neuronaux lors des prédictions. On peut trouver des pistes intéressantes dans la littérature récente visant le développement de systèmes plus explicatifs, notamment en analyse de sentiments via le concept d'aspect (Aspect Based Sentiment) qui donne une interprétation plus claire et plus précise [Silveira et al., 2019]. Il serait très intéressant d'en élargir la faisabilité à la classification de textes et également de proposer une découpe hiérarchique des aspects identifiés.

2.4 Visualisation interactive

Les modèles d'apprentissage profond sont difficilement interprétables par les spécialistes de l'apprentissage profond et particulièrement par les non spécialistes comme les professionnels de santé. La visualisation interactive du processus d'apprentissage des réseaux de neurones améliore cette intéprétabilité [Choo and Liu, 2018], malheureusement elle reste un domaine peu étudié encore. Cependant quelques études prometteuses ont été proposées par exemple [Shi et al., 2019]. Les auteurs proposent un framework DeepClue appliqué aux marchés financiers, système conçu pour relier les modèles d'apprentissage en profondeur textuels et les utilisateurs finaux en interprétant visuellement les facteurs clés appris dans le modèle de prédiction. Transposer cette étude au milieu de la santé serait très bénéfique à l'utilisation de ces systèmes par les non-spécialistes.

2.5 Interprétabilité du processus de prédiction

Enfin, à plus long terme, nos expérimentations montrent une forte variabilité de résultats suivant les jeux de données. L'analyse des textes par les réseaux de neurones actuels est très difficilement explicable formellement [Such et al., 2019]. Les décisions prisent par les réseaux de neurones restent abruptes et obscures. Concernant l'analyse de textes, elles ne sont pas en lien avec la grammaire ou la syntaxe d'une langue. Ces décisions sont prises en mettant simplement à jour certaines structures de construction des textes sans réaliser de réels raisonnements. Actuellement, le développement de graphes de neurones non organisés en couche avec des connexions inférentes commence à être étudié. L'association d'un graphe neuronal d'action et d'un graphe neuronal de connaissance [Vialatte, 2018] pourrait conduire aux prémices d'une pensée numérique dans le dessein de pouvoir généraliser et valider au mieux les connaissances acquises en phase d'entraînement. Il serait alors envisageable de pousser ces réseaux de neurones à penser par eux mêmes !

Bibliographie

Mohammad Reza Abbasifard, Bijan Ghahremani, and Hassan Naderi. Article : A survey on nearest neighbor search methods. *International Journal of Computer Applications*, 95(25) :39–52, June 2014. Full text available.

Muhammad Abulaish and Amit Kumar Sah. A text data augmentation approach for improving the performance of cnn. In *Proceedings of the 11th International Conference on Communication Systems & Networks*, COMSNETS 2019, 2019.

Khairul Anam and Adel Al-Jumaily. A novel extreme learning machine for dimensionality reduction on finger movement classification using semg. In *7th International IEEE/EMBS Conference on Neural Engineering, NER 2015, Montpellier, France, April 22-24, 2015*, pages 824–827. IEEE, 2015. doi: 10.1109/NER.2015.7146750. URL https://doi.org/10.1109/NER.2015.7146750.

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1 : Long Papers*. The Association for Computer Linguistics, 2016. doi: 10.18653/v1/p16-1231. URL https://doi.org/10.18653/v1/p16-1231.

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=ryghZJBKPS.

Moez Baccouche, Franck Mamalet, Christian Wolf, Christophe Garcia, and Atilla Baskurt. Sequential deep learning for human action recognition. In Albert Ali Salah and Bruno Lepri, editors, *Human Behavior Unterstanding - Second International Workshop, HBU 2011, Amsterdam, The Netherlands, November 16, 2011. Proceedings*, volume 7065 of *Lecture Notes in Computer Science*, pages 29–39. Springer, 2011. doi: 10.1007/978-3-642-25446-8_4. URL https://doi.org/10.1007/978-3-642-25446-8_4.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1409.0473.

Ruirui Bai, Zhongqing Wang, Fang Kong, Shoushan Li, and Guodong Zhou. Neural co-training for sentiment classification with product attributes. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 19(5), August 2020. ISSN 2375-4699. doi: 10.1145/3394113. URL https://doi.org/10.1145/3394113.

An Bang, Wenjun Wu, and Huimin Han. Deep active learning for text classification. In *Proceedings of the 2nd International Conference on Vision, Image and Signal Processing*, ICVISP 2018, pages 22 :1–22 :6, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-6529-1. doi: 10.1145/3271553.3271578. URL http://doi.acm.org/10.1145/3271553.3271578.

Clare Barrington, Cyprian Wejnert, Maria Elena Guardado, Ana Isabel Nieto, and Gabriela Paz Bailey. Social network characteristics and hiv vulnerability among transgender persons in san salvador : Identifying opportunities for hiv prevention strategies. *AIDS and Behavior*, 16(1) :214–224, Jan 2012. ISSN 1573-3254. doi: 10.1007/s10461-011-9959-1. URL https://doi.org/10.1007/s10461-011-9959-1.

Petr Belohlávek, Ondrej Plátek, Zdenek Zabokrtský, and Milan Straka. Using adversarial examples in natural language processing. In *Proceedings of the 11th International Conference on Language Resources and Evaluation, Miyazaki, Japan, May 7-12, 2018.*, LREC 2018, 2018. URL http://www.lrec-conf.org/proceedings/lrec2018/summaries/852.html.

Iz Beltagy, Kyle Lo, and Arman Cohan. Scibert : A pretrained language model for scientific text. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3613–3618. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1371. URL https://doi.org/10.18653/v1/D19-1371.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3 :1137–1155, 2003. URL http://jmlr.org/papers/v3/bengio03a.html.

Djallel Bouneffouf. Exponentiated gradient exploration for active learning. *Computers*, 5(1), 2014. URL http://arxiv.org/abs/1408.2196.

Leo Breiman. Random forests. *Mach. Learn.*, 45(1) :5–32, 2001. doi: 10.1023/A:1010933404324. URL https://doi.org/10.1023/A:1010933404324.

Leo Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984. ISBN 0-534-98053-8.

Mateusz Budnik. *Active and deep learning for multimedia. (Apprentissage actif et profond pour le multimédia)*. PhD thesis, Université Grenoble Alpes, 2017. URL https://tel.archives-ouvertes.fr/tel-01955048.

Imen Cheikhrouhou. *Description and classification of breast masses for the diagnosis of breast cancer*. Theses, Université d'Evry-Val d'Essonne, June 2012. URL https://tel.archives-ouvertes.fr/tel-00875976.

Shi-tao Chen, Songyi Zhang, Jinghao Shang, Badong Chen, and Nanning Zheng. Brain-inspired cognitive model with attention for self-driving cars. *IEEE Trans. Cogn. Dev. Syst.*, 11(1) :13–25, 2019. doi: 10.1109/TCDS.2017.2717451. URL https://doi.org/10.1109/TCDS.2017.2717451.

Tianqi Chen and Carlos Guestrin. Xgboost : A scalable tree boosting system. In Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi, editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 785–794. ACM, 2016. doi: 10.1145/2939672.2939785. URL https://doi.org/10.1145/2939672.2939785.

Jaegul Choo and Shixia Liu. Visual analytics for explainable deep learning. *CoRR*, abs/1804.02527, 2018. URL http://arxiv.org/abs/1804.02527.

Bee Bee Chua and Ying Zhang. Applying a systematic literature review and content analysis method to analyse open source developers' forking motivation interpretation, categories and consequences. *Australasian Journal of Information Systems*, 24, Jun. 2020. doi: 10.3127/ajis.v24i0.1714. URL https://journal.acs.org.au/index.php/ajis/article/view/1714.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12 :2493–2537, 2011. URL http://dl.acm.org/citation.cfm?id=2078186.

Michael Crawford, Taghi M. Khoshgoftaar, and Joseph D. Prusa. Reducing feature set explosion to facilitate real-world review spam detection. In Zdravko Markov and Ingrid Russell, editors, *Proceedings of the Twenty-Ninth International Florida Artificial Intelligence Research Society Conference, FLAIRS*

2016, Key Largo, Florida, USA, May 16-18, 2016, pages 304–309. AAAI Press, 2016. URL http://www.aaai.org/ocs/index.php/FLAIRS/FLAIRS16/paper/view/12844.

Aron Culotta and Andrew McCallum. Reducing labeling effort for structured prediction tasks. In Manuela M. Veloso and Subbarao Kambhampati, editors, *Proceedings, The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, July 9-13, 2005, Pittsburgh, Pennsylvania, USA*, pages 746–751. AAAI Press / The MIT Press, 2005. URL http://www.aaai.org/Library/AAAI/2005/aaai05-117.php.

Franck Dernoncourt and Ji Young Lee. Pubmed 200k RCT : a dataset for sequential sentence classification in medical abstracts. In Greg Kondrak and Taro Watanabe, editors, *Proceedings of the Eighth International Joint Conference on Natural Language Processing, IJCNLP 2017, Taipei, Taiwan, November 27 - December 1, 2017, Volume 2 : Short Papers*, pages 308–313. Asian Federation of Natural Language Processing, 2017. URL https://www.aclweb.org/anthology/I17-2052/.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT : pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1423. URL https://doi.org/10.18653/v1/n19-1423.

Quoc-Khanh Do, Alexandre Allauzen, and François Yvon. Apprentissage discriminant des modèles continus de traduction. In *Actes de la 22e conférence sur le Traitement Automatique des Langues Naturelles. Articles longs*, pages 267–278, Caen, France, June 2015. ATALA. URL https://www.aclweb.org/anthology/2015.jeptalnrecital-long.23.

Melanie Ducoffe and Frédéric Precioso. QBDC : query by dropout committee for training deep supervised architecture. *CoRR*, abs/1511.06412, 2015. URL http://arxiv.org/abs/1511.06412.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. Understanding back-translation at scale. *CoRR*, abs/1808.09381, 2018. URL http://arxiv.org/abs/1808.09381.

Charles Elkan. The foundations of cost-sensitive learning. In Bernhard Nebel, editor, *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI 2001, Seattle, Washington, USA, August 4-10, 2001*, pages 973–978. Morgan Kaufmann, 2001.

Caroline Etienne. *Deep learning applied to speech emotion recognition*. Theses, Université Paris-Saclay, December 2019. URL https://tel.archives-ouvertes.fr/tel-02479126.

Eduardo Fidalgo, Enrique Alegre, Laura Fernández-Robles, and Víctor González-Castro. Classifying suspicious content in tor darknet through semantic attention keypoint filtering. *Digital Investigation*, 30 :12–22, 2019. doi: 10.1016/j.diin.2019.05.004. URL https://doi.org/10.1016/j.diin.2019.05.004.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of online learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1) : 119–139, 1997. doi: 10.1006/jcss.1997.1504. URL https://doi.org/10.1006/jcss.1997.1504.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 1183–1192. JMLR.org, 2017. URL http://dl.acm.org/citation.cfm?id=3305381.3305504.

Shalini Ghosh, Oriol Vinyals, Brian Strope, Scott Roy, Tom Dean, and Larry P. Heck. Contextual LSTM (CLSTM) models for large scale NLP tasks. *CoRR*, abs/1602.06291, 2016. URL http://arxiv.org/abs/1602.06291.

Daniel Gissin and Shai Shalev-Shwartz. Discriminative active learning. *CoRR*, abs/1907.06347, 2019. URL http://arxiv.org/abs/1907.06347.

Evgin Göçeri. Analysis of deep networks with residual blocks and different activation functions : Classification of skin diseases. In *Ninth International Conference on Image Processing Theory, Tools and Applications, IPTA 2019, Istanbul, Turkey, November 6-9, 2019*, pages 1–6. IEEE, 2019. doi: 10.1109/IPTA.2019.8936083. URL https://doi.org/10.1109/IPTA.2019.8936083.

Felix Gräßer, Surya Kallumadi, Hagen Malberg, and Sebastian Zaunseder. Aspect-based sentiment analysis of drug reviews applying cross-domain and cross-data learning. In Patty Kostkova, Floriana Grasso, Carlos Castillo, Yelena Mejova, Arnold Bosman, and Michael Edelstein, editors, *Proceedings of the 2018 International Conference on Digital Health, DH 2018, Lyon, France, April 23-26, 2018*, pages 121–125. ACM, 2018. doi: 10.1145/3194658.3194677. URL https://doi.org/10.1145/3194658.3194677.

Hongyu Guo. Nonlinear mixup : Out-of-manifold data augmentation for text classification. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 4044–4051. AAAI Press, 2020. URL https://aaai.org/ojs/index.php/AAAI/article/view/5822.

Rahul Gupta. Data augmentation for low resource sentiment analysis using generative adversarial networks. *CoRR*, abs/1902.06818, 2019. URL http://arxiv.org/abs/1902.06818.

Changhee Han, Leonardo Rundo, Ryosuke Araki, Yujiro Furukawa, Giancarlo Mauri, Hideki Nakayama, and Hideaki Hayashi. Infinite brain MR images : Pggan-based data augmentation for tumor detection. *CoRR*, abs/1903.12564, 2019. URL http://arxiv.org/abs/1903.12564.

Nithin Haridas and Yubin Kim. Clustering large-scale diverse electronic medical records to aid annotation for generic named entity recognition. In Carsten Eickhoff, Yubin Kim, and Ryen W. White, editors, *Proceedings of the ACM WSDM 2020 Health Search and Data Mining Workshop, co-located with the 13th ACM International WSDM Conference, HSDM@WSDM 2020, Houston, TX, USA, February 3, 2020*, volume 2551 of *CEUR Workshop Proceedings*, pages 43–50. CEUR-WS.org, 2020. URL http://ceur-ws.org/Vol-2551/paper-08.pdf.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. Deep semantic role labeling : What works and what's next. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1 : Long Papers*, pages 473–483. Association for Computational Linguistics, 2017. doi: 10.18653/v1/P17-1044. URL https://doi.org/10.18653/v1/P17-1044.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015. URL http://arxiv.org/abs/1503.02531.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8) :1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

David J. Houghton, Andrew D. Pressey, and Doga Istanbulluoglu. Who needs social networking? an empirical enquiry into the capability of facebook to meet human needs and satisfaction with life. *Comput. Hum. Behav.*, 104 :106153, 2020. doi: 10.1016/j.chb.2019.09.029. URL https://doi.org/10.1016/j.chb.2019.09.029.

Jeremy Howard and Sebastian Ruder. Fine-tuned language models for text classification. *CoRR*, abs/1801.06146, 2018. URL http://arxiv.org/abs/1801.06146.

Rie Johnson and Tong Zhang. Effective use of word order for text categorization with convolutional neural networks. In Rada Mihalcea, Joyce Yue Chai, and Anoop Sarkar, editors, *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pages 103–112. The Association for Computational Linguistics, 2015. doi: 10.3115/v1/n15-1011. URL https://doi.org/10.3115/v1/n15-1011.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hervé Jégou, and Tomas Mikolov. Fasttext.zip : Compressing text classification models. *CoRR*, abs/1612.03651, 2016. URL http://arxiv.org/abs/1612.03651.

Kushal Kafle, Mohammed A. Yousefhussien, and Christopher Kanan. Data augmentation for visual question answering. In José M. Alonso, Alberto Bugarín, and Ehud Reiter, editors, *Proceedings of the 10th International Conference on Natural Language Generation, INLG 2017, Santiago de Compostela, Spain, September 4-7, 2017*, pages 198–202. Association for Computational Linguistics, 2017. doi: 10.18653/v1/w17-3529. URL https://doi.org/10.18653/v1/w17-3529.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1 : Long Papers*, pages 655–665. The Association for Computer Linguistics, 2014. doi: 10.3115/v1/p14-1062. URL https://doi.org/10.3115/v1/p14-1062.

Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. A large self-annotated corpus for sarcasm. In Nicoletta Calzolari, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Kôiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asunción Moreno, Jan Odijk, Stelios Piperidis, and Takenobu Tokunaga, editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*. European Language Resources Association

(ELRA), 2018. URL http://www.lrec-conf.org/proceedings/lrec2018/summaries/160.html.

Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751, 2014. URL http://aclweb.org/anthology/D/D14/D14-1181.pdf.

Sosuke Kobayashi. Contextual augmentation : Data augmentation by words with paradigmatic relations. In Marilyn A. Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 452–457. Association for Computational Linguistics, 2018. doi: 10.18653/v1/n18-2072. URL https://doi.org/10.18653/v1/n18-2072.

Jonathan Kobold. *Deep Learning for lesion and thrombus segmentation from cerebral MRI*. PhD thesis, Paris-Saclay, 2019. URL http://www.theses.fr/2019SACLE044. Thèse de doctorat dirigée par Maaref, Hichem et Vigneron, Vincent Traitement du signal et des images Université Paris-Saclay (ComUE) 2019.

Bartosz Krawczyk, Bridget T. McInnes, and Alberto Cano. Sentiment classification from multi-class imbalanced twitter data using binarization. In Francisco Javier Martínez de Pisón, Rubén Urraca, Héctor Quintián, and Emilio Corchado, editors, *Hybrid Artificial Intelligent Systems - 12th International Conference, HAIS 2017, La Rioja, Spain, June 21-23, 2017, Proceedings*, volume 10334 of *Lecture Notes in Computer Science*, pages 26–37. Springer, 2017. doi: 10.1007/978-3-319-59650-1_3. URL https://doi.org/10.1007/978-3-319-59650-1_3.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25 : 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 1106–1114, 2012. URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.

Taku Kudo and John Richardson. Sentencepiece : A simple and language independent subword tokenizer and detokenizer for neural text processing. In

Eduardo Blanco and Wei Lu, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018 : System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*, pages 66–71. Association for Computational Linguistics, 2018. doi: 10.18653/v1/d18-2012. URL https://doi.org/10.18653/v1/d18-2012.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. Ask me anything : Dynamic memory networks for natural language processing. In Maria-Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1378–1387. JMLR.org, 2016. URL http://proceedings.mlr.press/v48/kumar16.html.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. Recurrent convolutional neural networks for text classification. In *AAAI*, 2015.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT : A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942, 2019. URL http://arxiv.org/abs/1909.11942.

Hoa T. Le, Christophe Cerisara, and Alexandre Denis. Do convolutional networks need to be deep for text classification? *CoRR*, abs/1707.04108, 2017. URL http://arxiv.org/abs/1707.04108.

Jerome Lettvin, Humberto Maturana, Warren McCulloch, and Walter Pitts. What the frog's eye tells the frog's brain. *Proceedings of the Institute of Radic Engineers*, 47 :1940–1959, 1959. URL https://neuromajor.ucr.edu/courses/WhatTheFrogsEyeTellsTheFrogsBrain.pdf.

David D. Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the 11th International Conference on Machine Learning*, ICML 1994, pages 148–156. Morgan Kaufmann, 1994.

Depeng Liang and Yongdong Zhang. AC-BLSTM : asymmetric convolutional bidirectional LSTM networks for text classification. *CoRR*, abs/1611.01884, 2016. URL http://arxiv.org/abs/1611.01884.

Zachary Chase Lipton. The mythos of model interpretability. *CoRR*, abs/1606.03490, 2016. URL http://arxiv.org/abs/1606.03490.

Weibo Liu, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, and Fuad E. Alsaadi. A survey of deep neural network architectures and their applications. *Neurocomputing*, 234 :11–26, 2017. doi: 10.1016/j.neucom.2016.12.038. URL https://doi.org/10.1016/j.neucom.2016.12.038.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta : A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019. URL http://arxiv.org/abs/1907.11692.

Ramón Maldonado and Sanda M. Harabagiu. Active deep learning for the identification of concepts and relations in electroencephalography reports. *J. Biomed. Informatics*, 98, 2019. doi: 10.1016/j.jbi.2019.103265. URL https://doi.org/10.1016/j.jbi.2019.103265.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. Learned in translation : Contextualized word vectors. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30 : Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 6294–6305, 2017. URL http://papers.nips.cc/paper/7209-learned-in-translation-contextualized-word-vectors.

David McClosky, Eugene Charniak, and Mark Johnson. Effective self-training for parsing. In Robert C. Moore, Jeff A. Bilmes, Jennifer Chu-Carroll, and Mark Sanderson, editors, *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 4-9, 2006, New York, New York, USA*. The Association for Computational Linguistics, 2006. URL https://www.aclweb.org/anthology/N06-1020/.

Yves Mercadier, Jérôme Azé, Sandra Bringay, Viviane Clavier, Erick Cuenca, Céline Paganelli, Pascal Poncelet, and Arnaud Sallaberry. #aids analyse information dangers sexualité : caractériser les discours à propos du VIH dans les forums de santé. In Sylvie Ranwez, editor, *IC 2018 : 29es Journées francophones d'Ingénierie des Connaissances (Proceedings of the 29th French Knowledge Engineering Conference), Nancy, France, July 4-6, 2018*, pages 71–86, 2018. URL https://hal.archives-ouvertes.fr/IC_2018/hal-01839552.

Yves Mercadier, Bilel Moulahi, Sandra Bringay, Jérôme Azé, Philippe Lenoir, Grégoire Mercier, and François Carbonnel. *Analysis by Multiclass Multilabel Classification of the 2015 #SmearForSmear Campaign Using Deep Learning*,

pages 193–205. Springer International Publishing, Cham, 2019. ISBN 978-3-030-14714-3. URL https://doi.org/10.1007/978-3-030-14714-3_10.

Yves Mercadier, Jérôme Azé, and Sandra Bringay. Apprentissage actif profond pour le classement de textes en plusieurs classes. In Antoine Cornuéjols and Etienne Cuvelier, editors, *Extraction et Gestion des Connaissances, EGC 2020, Brussels, Belgium, January 27-31, 2020*, volume E-36 of *RNTI*, pages 49–60. Éditions RNTI, 2020a. URL http://editions-rnti.fr/?inprocid=1002561.

Yves Mercadier, Jérôme Azé, and Sandra Bringay. Divide to better classify. In Martin Michalowski and Robert Moskovitch, editors, *Artificial Intelligence in Medicine - 18th International Conference on Artificial Intelligence in Medicine, AIME 2020, Minneapolis, MN, USA, August 25-28, 2020, Proceedings*, volume 12299 of *Lecture Notes in Computer Science*, pages 89–99. Springer, 2020b. doi: 10.1007/978-3-030-59137-3_9. URL https://doi.org/10.1007/978-3-030-59137-3_9.

Yves Mercadier, Sandra Bringay, and Jérôme Azé. *Divide to better classify*, chapter 9. Springer Nature, 7th Floor Block - C Hardy Tower Ramanujan IT City Rajiv Gandhi Salai (OMR) Taramani Chennai - 600 113 India, 2021.

Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In Takao Kobayashi, Keikichi Hirose, and Satoshi Nakamura, editors, *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, pages 1045–1048. ISCA, 2010. URL http://www.isca-speech.org/archive/interspeech_2010/i10_1045.html.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26 : 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119, 2013a. URL http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedins of the Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT-NAACL 2013, pages 746–751, 2013b.

Dat Ba Nguyen, Martin Theobald, and Gerhard Weikum. J-NERD : joint named entity recognition and disambiguation with rich linguistic features. *Trans. Assoc. Comput. Linguistics*, 4 :215–229, 2016. URL https://tacl2013.cs.columbia.edu/ojs/index.php/tacl/article/view/698.

Niki Parmar, Prajit Ramachandran, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-alone self-attention in vision models. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32 : Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 68–80, 2019. URL http://papers.nips.cc/paper/8302-stand-alone-self-attention-in-vision-models.

Yifan Peng, Shankai Yan, and Zhiyong Lu. Transfer learning in biomedical natural language processing : An evaluation of BERT and elmo on ten benchmarking datasets. *CoRR*, abs/1906.05474, 2019. URL http://arxiv.org/abs/1906.05474.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove : Global vectors for word representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL, 2014. doi: 10.3115/v1/d14-1162. URL https://doi.org/10.3115/v1/d14-1162.

Luis Perez and Jason Wang. The effectiveness of data augmentation in image classification using deep learning. *CoRR*, abs/1712.04621, 2017. URL http://arxiv.org/abs/1712.04621.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In Marilyn A. Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics, 2018. doi: 10.18653/v1/n18-1202. URL https://doi.org/10.18653/v1/n18-1202.

N. C. Phuong and T. D. Dat. Sound classification for event detection : Application into medical telemonitoring. In *2013 International Conference on Computing, Management and Telecommunications (ComManTel)*, pages 330–333, 2013.

Waleed Ragheb, Bilel Moulahi, Jérôme Azé, Sandra Bringay, and Maximilien Servajean. Temporal mood variation : at the CLEF erisk-2018 tasks for early risk detection on the internet. In Linda Cappellato, Nicola Ferro, Jian-Yun Nie, and Laure Soulier, editors, *Working Notes of CLEF 2018 - Conference and Labs of the Evaluation Forum, Avignon, France, September 10-14, 2018*, volume 2125 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2018. URL http://ceur-ws.org/Vol-2125/paper_78.pdf.

Waleed Ragheb, Jérôme Azé, Sandra Bringay, and Maximilien Servajean. Attentive multi-stage learning for early risk detection of signs of anorexia and self-harm on social media. In Linda Cappellato, Nicola Ferro, David E. Losada, and Henning Müller, editors, *Working Notes of CLEF 2019 - Conference and Labs of the Evaluation Forum, Lugano, Switzerland, September 9-12, 2019*, volume 2380 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019. URL http://ceur-ws.org/Vol-2380/paper_126.pdf.

Jason D. M. Rennie, Lawrence Shih, Jaime Teevan, and David R. Karger. Tackling the poor assumptions of naive bayes text classifiers. In *In Proceedings of the Twentieth International Conference on Machine Learning*, pages 616–623, 2003.

Aude Robert, Yasmine Baghdadi, Pierre Zweigenbaum, Claire Morgand, Cyril Grouin, Thomas Lavergne, Aurélie Névéol, Anne Fouillet, and G. Rey. Développement et application de méthodes de traitement automatique des langues sur les causes médicales de décès pour la santé publique. *Bulletin Epidémiologique Hebdomadaire - BEH*, 5 :603–609, 2019. URL https://hal.archives-ouvertes.fr/hal-02411625.

Xin Rong. word2vec parameter learning explained. *CoRR*, abs/1411.2738, 2014. URL http://arxiv.org/abs/1411.2738.

Sara Rosenthal, Saif M. Mohammad, Preslav Nakov, Alan Ritter, Svetlana Kiritchenko, and Veselin Stoyanov. Semeval-2015 task 10 : Sentiment analysis in twitter. *CoRR*, abs/1912.02387, 2019. URL http://arxiv.org/abs/1912.02387.

Dmitri Roussinov and J. Leon Zhao. Text clustering and summary techniques for CRM message management. *J. Enterp. Inf. Manag.*, 17(6) :424–429, 2004. doi: 10.1108/17410390410566715. URL https://doi.org/10.1108/17410390410566715.

Gerard Salton and Chris Buckley. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manag.*, 24(5) :513–523, 1988. doi: 10.1016/0306-4573(88)90021-0. URL https://doi.org/10.1016/0306-4573(88)90021-0.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT : smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019. URL http://arxiv.org/abs/1910.01108.

Tobias Scheffer, Christian Decomain, and Stefan Wrobel. Active hidden markov models for information extraction. In Frank Hoffmann, David J. Hand, Niall Adams, Douglas Fisher, and Gabriela Guimaraes, editors, *Advances in Intelligent Data Analysis*, pages 309–318, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg. ISBN 978-3-540-44816-7.

Jürgen Schmidhuber. Deep learning in neural networks : An overview. *CoRR*, abs/1404.7828, 2014. URL http://arxiv.org/abs/1404.7828.

Holger Schwenk. Continuous space language models. *Comput. Speech Lang.*, 21(3) :492–518, 2007. doi: 10.1016/j.csl.2006.09.003. URL https://doi.org/10.1016/j.csl.2006.09.003.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks : A core-set approach. In *Proceedings of the 6th International Conference on Learning Representations, Vancouver, BC, Canada, April 30 - May 3, 2018*, ICLR 2018, 2018. URL https://openreview.net/forum?id=H1aIuk-RW.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. *CoRR*, abs/1511.06709, 2015. URL http://arxiv.org/abs/1511.06709.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009. URL http://axon.cs.byu.edu/~martinez/classes/778/Papers/settles.activelearning.pdf.

Claude E. Shannon. A mathematical theory of communication. *Mobile Computing and Communications Review*, 5(1) :3–55, 2001. doi: 10.1145/584091.584093. URL https://doi.org/10.1145/584091.584093.

G. L. Shaw. Donald hebb : The organization of behavior. In Günther Palm and Ad Aertsen, editors, *Brain Theory*, pages 231–233, Berlin, Heidelberg, 1986. Springer Berlin Heidelberg. ISBN 978-3-642-70911-1.

Yanyao Shen, Hyokun Yun, Zachary Chase Lipton, Yakov Kronrod, and Animashree Anandkumar. Deep active learning for named entity recognition. In *Proceedings of the 2nd Workshop on Representation Learning for NLP, Vancouver, Canada, August 3, 2017*, Rep4NLP@ACL 2017, pages 252–256, 2017. URL https://www.aclweb.org/anthology/W17-2630/.

Lei Shi, Zhiyang Teng, Le Wang, Yue Zhang, and Alexander Binder. Deepclue : Visual interpretation of text-based deep stock prediction. *IEEE Trans. Knowl. Data Eng.*, 31(6) :1094–1108, 2019. doi: 10.1109/TKDE.2018.2854193. URL https://doi.org/10.1109/TKDE.2018.2854193.

Sam Shleifer. Low resource text classification with ulmfit and backtranslation. *CoRR*, abs/1903.09244, 2019. URL http://arxiv.org/abs/1903.09244.

Connor Shorten and Taghi M. Khoshgoftaar. A survey on image data augmentation for deep learning. *J. Big Data*, 6 :60, 2019. doi: 10.1186/s40537-019-0197-0. URL https://doi.org/10.1186/s40537-019-0197-0.

Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *CoRR*, abs/1703.00810, 2017. URL http://arxiv.org/abs/1703.00810.

Aditya Siddhant and Zachary C. Lipton. Deep bayesian active learning for natural language processing : Results of a large-scale empirical study. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, EMNLP 2018, pages 2904–2909, 2018. URL https://www.aclweb.org/anthology/D18-1318/.

Thiago De Sousa Silveira, Hans Uszkoreit, and Renlong Ai. Using aspect-based analysis for explainable sentiment predictions. In Jie Tang, Min-Yen Kan, Dongyan Zhao, Sujian Li, and Hongying Zan, editors, *Natural Language Processing and Chinese Computing - 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9-14, 2019, Proceedings, Part II*, volume 11839 of *Lecture Notes in Computer Science*, pages 617–627. Springer, 2019. doi: 10.1007/978-3-030-32236-6_56. URL https://doi.org/10.1007/978-3-030-32236-6_56.

Edwin Simonnet. *Deep learning applied to spoken langage understanding*. Theses, Université du Maine, February 2019. URL https://tel.archives-ouvertes.fr/tel-02077011.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. ACL, 2013. URL https://www.aclweb.org/anthology/D13-1170/.

Mengjiao Song, Xingyu Zhao, Yong Liu, and Zhihong Zhao. Text sentiment analysis based on convolutional neural network and bidirectional LSTM model. In Qinglei Zhou, Qiguang Miao, Hongzhi Wang, Wei Xie, Yan Wang, and Zeguang Lu, editors, *Data Science - 4th International Conference of Pioneering Computer Scientists, Engineers and Educators, ICPCSEE 2018, Zhengzhou, China, September 21-23, 2018, Proceedings, Part II*, volume 902 of *Communications in Computer and Information Science*, pages 55–68. Springer, 2018. doi: 10.1007/978-981-13-2206-8_6. URL https://doi.org/10.1007/978-981-13-2206-8_6.

Marijn F Stollenga, Wonmin Byeon, Marcus Liwicki, and Jürgen Schmidhuber. Parallel multi-dimensional lstm, with application to fast biomedical volumetric image segmentation. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2998–3006. Curran Associates, Inc., 2015. URL http://papers.nips.cc/paper/5642-parallel-multi-dimensional-lstm-with-application-to-fast-biomedical-volumetric-image-segmentation.pdf.

Ondrej Such, Peter Tarábek, Katarína Bachratá, and Andrea Tinajová. Pairwise coupling of convolutional neural networks for better explicability of classification systems. *CoRR*, abs/1911.03645, 2019. URL http://arxiv.org/abs/1911.03645.

Amane Sugiyama and Naoki Yoshinaga. Data augmentation using back-translation for context-aware neural machine translation. In *Proceedings of the Fourth Workshop on Discourse in Machine Translation (DiscoMT 2019)*, pages 35–44, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-6504. URL https://www.aclweb.org/anthology/D19-6504.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28 : Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2440–2448, 2015. URL http://papers.nips.cc/paper/5846-end-to-end-memory-networks.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017a. URL http://arxiv.org/abs/1706.03762.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30 : Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008, 2017b. URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.

Jean-Charles Vialatte. *On convolution of graph signals and deep learning on graph domains. (Convolution et apprentissage profond sur graphes)*. PhD thesis, École nationale supérieure Mines-Télécom Atlantique, France, 2018. URL https://tel.archives-ouvertes.fr/tel-02136338.

Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey E. Hinton. Grammar as a foreign language. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28 : Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2773–2781, 2015. URL http://papers.nips.cc/paper/5635-grammar-as-a-foreign-language.

Yasi Wang, Hongxun Yao, and Sicheng Zhao. Auto-encoder based dimensionality reduction. *Neurocomputing*, 184 :232 – 242, 2016. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2015.08.104. URL http://www.sciencedirect.com/science/article/pii/S0925231215017671. RoLoD : Robust Local Descriptors for Computer Vision 2014.

Zheng Wang and Jieping Ye. Querying discriminative and representative samples for batch mode active learning. In *Proceedings of the 19th ACM International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 158–166, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2174-7. doi: 10.1145/2487575.2487643. URL http://doi.acm.org/10.1145/2487575.2487643.

Jason W. Wei and Kai Zou. EDA : easy data augmentation techniques for boosting performance on text classification tasks. *CoRR*, abs/1901.11196, 2019. URL http://arxiv.org/abs/1901.11196.

Ho Chung Wu, Robert Wing Pong Luk, Kam-Fai Wong, and Kui-Lam Kwok. Interpreting TF-IDF term weights as making relevance decisions. *ACM Trans. Inf. Syst.*, 26(3) :13 :1–13 :37, 2008. doi: 10.1145/1361684.1361686. URL http://doi.acm.org/10.1145/1361684.1361686.

Ting-Fan Wu, Chih-Jen Lin, and Ruby C. Weng. Probability estimates for multi-class classification by pairwise coupling. *J. Mach. Learn. Res.*, 5 :975–1005, 2004. URL http://jmlr.org/papers/volume5/wu04a/wu04a.pdf.

Qizhe Xie, Zihang Dai, Eduard H. Hovy, Minh-Thang Luong, and Quoc V. Le. Unsupervised data augmentation. *CoRR*, abs/1904.12848, 2019. URL http://arxiv.org/abs/1904.12848.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. BERT post-training for review reading comprehension and aspect-based sentiment analysis. *CoRR*, abs/1904.02232, 2019. URL http://arxiv.org/abs/1904.02232.

Jiaming Xu, Bo Xu, Peng Wang, Suncong Zheng, Guanhua Tian, and Jun Zhao. Self-taught convolutional neural networks for short text clustering. *Neural Networks*, 88 :22–31, 2017. doi: 10.1016/j.neunet.2016.12.008. URL https://doi.org/10.1016/j.neunet.2016.12.008.

Rikiya Yamashita, Mizuho Nishio, Richard K. G. Do, and Kaori Togashi. Convolutional neural networks : an overview and application in radiology. *Insights into Imaging*, 9 :611 – 629, 2018.

Bishan Yang, Jian-Tao Sun, Tengjiao Weng, and Zheng Chen. Effective multi-label active learning for text classification. In *The 15th ACM SIGKDD Conference On Knowledge Discovery and Data Mining*. Association for Computing Machinery, Inc., January 2009. URL https://www.microsoft.com/en-us/research/publication/effective-multi-label-active-learning-for-text-classification/.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet : Generalized autoregressive pretraining for language understanding. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32 : Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 5754–5764, 2019. URL http://papers.nips.cc/paper/8812-xlnet-generalized-autoregressive-pretraining-for-language-understanding.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. Qanet : Combining local convolution with global self-attention for reading comprehension. *CoRR*, abs/1804.09541, 2018. URL http://arxiv.org/abs/1804.09541.

Sergey Zagoruyko and Nikos Komodakis. Diracnets : Training very deep neural networks without skip-connections. *CoRR*, abs/1706.00388, 2017. URL http://arxiv.org/abs/1706.00388.

Biao Zhang, Deyi Xiong, Jinsong Su, Hong Duan, and Min Zhang. Bilingual autoencoders with global descriptors for modeling parallel sentences. In Nicoletta Calzolari, Yuji Matsumoto, and Rashmi Prasad, editors, *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference : Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 2548–2558. ACL, 2016a. URL https://www.aclweb.org/anthology/C16-1240/.

Xiang Zhang and Yann LeCun. Text understanding from scratch. *CoRR*, abs/1502.01710, 2015. URL http://arxiv.org/abs/1502.01710.

Ye Zhang, Stephen Roller, and Byron C. Wallace. MGNC-CNN : A simple approach to exploiting multiple word embeddings for sentence classification. *CoRR*, abs/1603.00968, 2016b. URL http://arxiv.org/abs/1603.00968.

Ye Zhang, Matthew Lease, and Byron C. Wallace. Active discriminative text representation learning. In *Proceedings of the 31th Conference on Artificial Intelligence*, AAAI'17, pages 3386–3392. AAAI Press, 2017. URL http://dl.acm.org/citation.cfm?id=3298023.3298060.

Ziqi Zhang and Monica Paramita. Product classification using microdata annotations. In Chiara Ghidini, Olaf Hartig, Maria Maleshkova, Vojtech Svátek, Isabel F. Cruz, Aidan Hogan, Jie Song, Maxime Lefrançois, and Fabien Gandon, editors, *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part I*, volume 11778 of *Lecture Notes in Computer Science*, pages 716–732. Springer, 2019. doi: 10.1007/978-3-030-30793-6_41. URL https://doi.org/10.1007/978-3-030-30793-6_41.

Sizheng Steven Zhao, Chuan Hong, Tianrun Cai, Chang Xu, Jie Huang, Joerg Ermann, Nicola J Goodson, Daniel H Solomon, Tianxi Cai, and Katherine P Liao. Incorporating natural language processing to improve classification of axial spondyloarthritis using electronic health records. *Rheumatology*, 59(5) : 1059–1065, 09 2019. ISSN 1462-0324. doi: 10.1093/rheumatology/kez375. URL https://doi.org/10.1093/rheumatology/kez375.

Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis C. M. Lau. A C-LSTM neural network for text classification. *CoRR*, abs/1511.08630, 2015. URL http://arxiv.org/abs/1511.08630.

Muhua Zhu, Yue Zhang, Wenliang Chen, Min Zhang, and Jingbo Zhu. Fast and accurate shift-reduce constituent parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1 : Long Papers*, pages 434–443. The Association for Computer Linguistics, 2013. URL https://www.aclweb.org/anthology/P13-1043/.

RNN

$MCOM$ Matrice de Co-Occurence des Mots.

n Taille du vocabulaire.

AB An AdaBoost classifier.

ABLSTM Attention Bidirectional Long short-term memory.

ACLSTM Asymmetric Convolutional Bidirectional Long short-term memory.

ALBERT A Lite BERT for Self-supervised Learning of Language Representations.

BERT Bidirectional Encoder Representations from Transformers.

BERT Bidirectional Encoder Representations from Transformers

BLSTM Bidirectional Long short-term memory.

CBOW continuous bag of words.

CLSTM Convolution Long short-term memory.

CNB Naive Bayes classifier for multivariate Bernoulli models.

CNN Convolutional neural network.

CNN2D Convolutional neural network 2 Dimensions.

COVE Contextualized Word Vectors.

CRM Customer Relationship Management

d vecteur représentation numérique d'un document.

DAL Discriminative Active Learning

DistilBERT Bidirectional Long short-term memory.

DT A decision tree classifier.

ELMo Embeddings from Language Models

GAFAM Google, Apple, Facebook, Amazon et Microsoft

GAN Generative Adversarial Network

GLUE General Language Understanding Evaluation.

KNN Classifier implementing the k-nearest neighbors vote.

LSTM Long short-term memory.

MGNCCNN Multi-Group Norm Constraint Convolutional Neural Network.

MLM Masked Language Modeling

NBIC nanotechnologies, biotechnologies, informatique, et sciences cognitives

NSP Next Sentence Prédiction

ReLu Rectified Linear unit

RF A random forest classifier.

RoBERTa A Robustly Optimized BERT Pretraining Approach.

ROC Receiver Operating Characteristic

SciBERT A Pretrained Language Model for Scientific Text.

SOP Sentence-Order Prediction

SVM Support Vector Machines.

SVM Séparateur à Vaste Marge.

TAL Traitement Automatique de la Langue

UMLFIT Universal Language Model Fine-tuning for Text Classification

VC Méta Classifier Commettee.

XGB eXtreme Gradient Boosting.

XLNet Generalized Autoregressive Pretraining for Language Understanding.

Annexes

A

Exploration des données

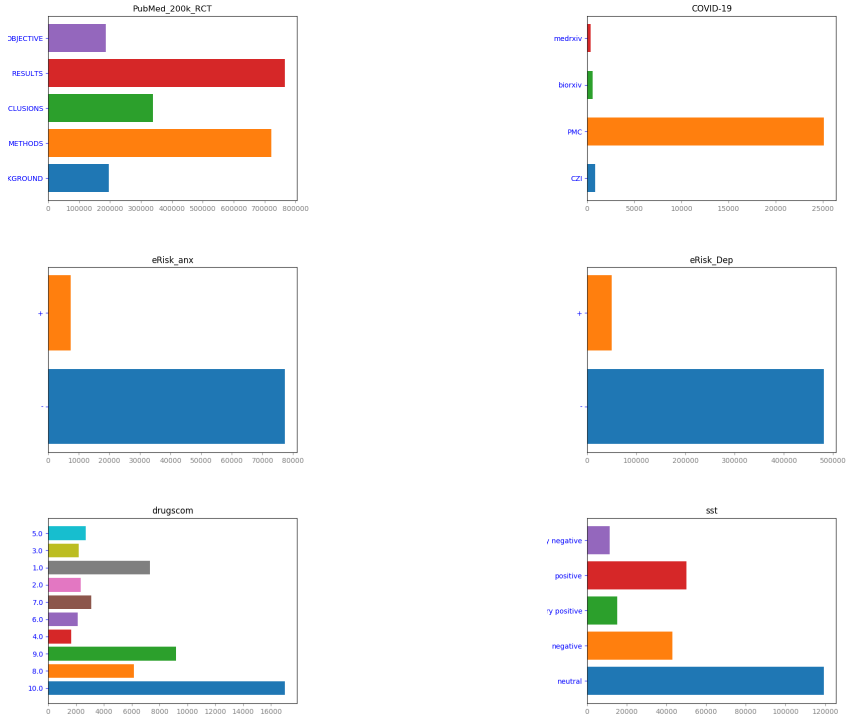
Nous faisons le choix de ces ensembles de texte car pour la grande majorité d'entre eux, ils sont utilisés par la communauté scientifique. De plus, ils correspondent également à un panel de tâches spécifiques couramment étudié dans le domaine de la santé. En outre, le nombre de classes de ces jeux de données variant de un à dix est de surcroît opportun dans l'idée de produire des études comparatives.

1 Analyse des jeux de données

1.1 Déséquilibre des classes

Nous pouvons constater dans la table A.2 que l'ensemble des jeux de données sont déséquilibrés en nombre d'échantillons par classe. Dans ces conditions, il faut bien appréhender la métrique d'évaluation utilisée, l'exactitude que nous précisons au paragraphe 2.3, pour évaluer les méthodes de classification. De fait, l'utilisation de l'exactitude doit être bien comprise. Prenons l'exemple du jeux de données eRisk anorexie, nous voyons très clairement un déséquilibre de classe. La classe négative détient environ 75 000 échantillons là où la classe positive contient environ 10 000 textes. Dans l'hypothèse où le classifieur prédit uniquement la classe négative pour tout échantillon, la mesure d'exactitude donne 0.8823%. Cette valeur peut paraître très bonne alors qu'elle est triviale à obtenir !

FIGURE A.1 – Déséquilibre des classes



1.2 Déséquilibre de la taille des documents

On peut remarquer sur le tableau A.2 que les jeux de données étudiés ne sont pas homogènes en nombre de mots par document. Cela peut avoir une forte incidence sur les prédictions produites par un classifieur. En effet, les classifieurs numériques utilisent des représentations de textes à taille fixe que nous présentons 1.1.

Nous notons également un déséquilibre de longueur de document en nombre de mots par classe de documents table A.3. Cela peut sembler anodin cependant, il'en est rien. Les modèles de classifieurs utilisés pour une étude ne peuvent fonctionner qu'avec un nombre de mots fixé par document. Aussi le choix de ce nombre de mots est primordial afin optimiser la performance du classifieur.

FIGURE A.2 – Longueur en nombre de mots des documents

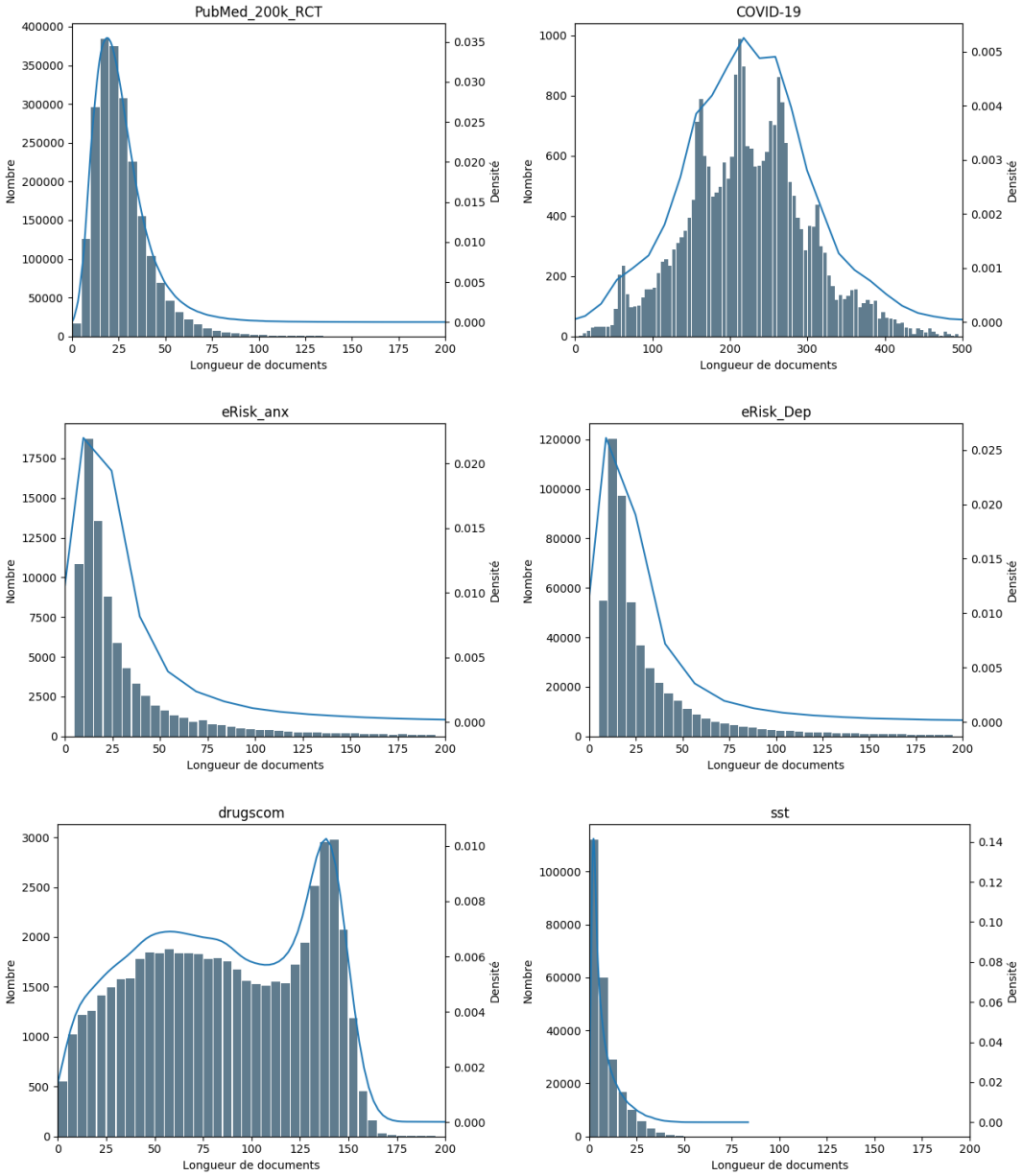
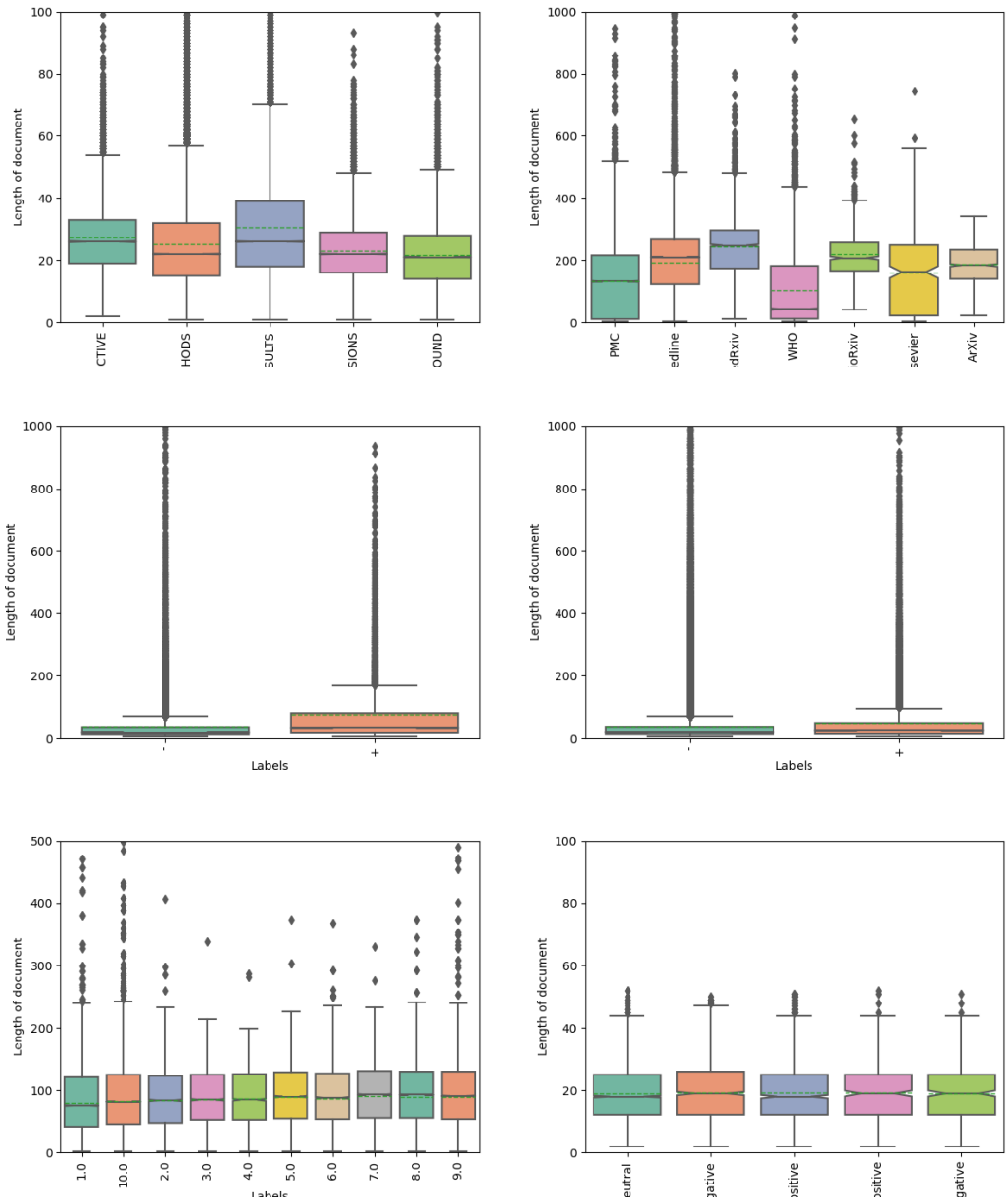


FIGURE A.3 – Longueur en nombre de mots par classe



En fin de compte, ces ensembles de textes sont fortement variés en termes de nombre de classes, de nombre de documents par classe, de nombre de mots par texte et de complexité.