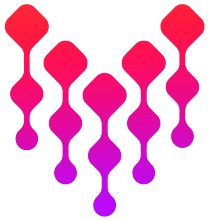
Rapport de stage, département Génie Informatique
(TN09)

Traitement automatique du langage pour la recherche de prospect sur les réseaux sociaux

Stage assistant ingénieur chez Makezu du 01 septembre 2020 au 12 février 2021



MAKEZU.IO

Adresse : 35 Rue du Sentier, 75002 Paris
Auteur : Gustave Cortal

Suiveur entreprise : H. Lucien
Suiveur UTC : D. Lourdeaux

Semestre d'automne 2020

Remerciements

Pour mon stage d'assistant-ingénieur, j'ai eu l'opportunité de rejoindre l'entreprise Makezu. Je tiens tout d'abord à remercier Hélène et Hugo qui m'ont fait confiance en me recrutant en tant que développeur intelligence artificielle. Malgré les conditions sanitaires qui m'ont obligé à effectuer une partie de mon stage à distance, la coopération avec Hélène et Hugo s'est effectuée parfaitement, grâce à leur professionnalisme et leur confiance en mon travail. Je tiens tout particulièrement à souligner la part importante qui est attribuée à la créativité, ce qui a permis de m'épanouir pleinement au sein des différents projets. Je tiens également à remercier Domitile Lourdeaux pour son encadrement durant le stage.

Sommaire

Résumé technique **3**

1 Présentation de Makezu **4**

1.1 Présentation de l'entreprise 4

1.2 Présentation de l'équipe 6

2 Amélioration et évolution des algorithmes Makezu **8**

2.1 Amélioration de l'application et évolution des algorithmes d'intelligence artificielle . 8

2.2 Contributions 9

2.3 Environnement technologique 9

2.4 Planning 10

2.5 Prise de recul 11

3 Extraction et automatisation du contenu sur les différentes plateformes sociales **13**

3.1 Quora 13

3.2 Twitter 16

3.3 LinkedIn 22

4 Catégorisation des textes pour l'analyse des tendances et le profilage des utilisateurs sur Twitter **23**

4.1 Introduction 23

4.2 Modèle de classification des textes en anglais 25

4.3 Élaboration d'un jeu de données composé d'articles francophones de journaux en ligne 29

4.4 Modèle de classification des textes en français 33

4.5 Mise en production des modèles pour Makezu 35

Conclusion **38**

Acronymes **46**

Glossaire **47**

Résumé technique

Le stage a permis d'améliorer l'application de l'entreprise reposant sur du traitement automatique du langage naturel (NLP ou *Natural Language Processing*). Le NLP est un domaine multidisciplinaire impliquant la linguistique, l'informatique et les statistiques. Il vise à créer des outils de traitement de la langue naturelle pour diverses applications comme l'analyse de sentiment, la catégorisation de documents ou la reconnaissance d'entités nommées. Nous avons construit des programmes permettant d'automatiser le contenu des clients pour qu'ils puissent plus facilement construire une communauté sur les réseaux sociaux, notamment Twitter. De plus, nous avons élargi le champ d'action de l'application Makezu en permettant l'extraction des données sur d'autres plateformes sociales comme Quora. Ce rapport commence par présenter Makezu et son équipe et explique la mission générale du stagiaire. Ensuite, les méthodes d'extraction et d'automatisation du contenu sur les différentes plateformes sociales sont expliquées. Nous finissons par présenter l'élaboration de modèles de classification permettant d'analyser les tendances Twitter et de profiler les utilisateurs grâce à la catégorisation du contenu textuel.

Chapitre 1

Présentation de Makezu

1.1 Présentation de l'entreprise

1.1.1 Brève histoire de l'entreprise

Makezu est une jeune start-up parisienne, fondée en novembre 2019, par deux jeunes entrepreneuses : Hélène Lucien, forte d'une première expérience dans la création de start-up autour de Twitter, et Anne-Sophie De Gabriac, qui partage son temps entre Makezu et une thèse. L'objectif de mon stage est d'améliorer l'outil de traitement automatique du langage, en collaboration avec Hugo Lacauste, ancien stagiaire devenu responsable de l'intelligence artificielle. Le projet de Makezu est de permettre à des entreprises de toutes tailles et de tous moyens de gagner de la visibilité sur les réseaux sociaux en construisant une communauté. À mon arrivée, Makezu s'était concentré principalement sur la plateforme Twitter. Cette plateforme permet d'envoyer gratuitement des micromessages appelés tweets limités à 280 caractères. C'est un réseau social très intéressant pour le marketing puisque les utilisateurs n'hésitent pas à partager leur vie quotidienne ainsi que leurs ressentis. De ce constat, ainsi que d'une collaboration avec l'école CentraleSupElec, est née l'application Makezu. Celle-ci met en relation des personnes sur Twitter exprimant un besoin avec les utilisateurs de l'application Makezu qui cherchent des clients potentiels. L'application repose sur de l'apprentissage automatique, qui se fonde sur des approches statistiques pour donner aux ordinateurs la capacité d'« apprendre » à partir de données. Nous verrons dans la prochaine sous-section plus de détails sur le fonctionnement de l'application.

1.1.2 Makezu et les campagnes

Makezu fonctionne sous forme de campagne. Une campagne lancée sur Makezu peut avoir trois buts différents :

— le client peut cibler un besoin. Si par exemple notre client est un vendeur de voiture, celui-ci peut cibler les personnes exprimant leur besoin d'acquérir une nouvelle voiture ;
— le client peut cibler un moment de vie clé. Les moments de vie correspondent à des moments significatifs dans la vie d'une personne, comme un décès, un mariage, une grossesse ou un anniversaire ;
— le client peut choisir de construire une communauté. Il peut cibler des personnes qui se définissent en « tant que X ». Par exemple, si notre client est un vendeur de produit bio, il peut choisir de cibler des personnes qui se définissent en tant que végan.

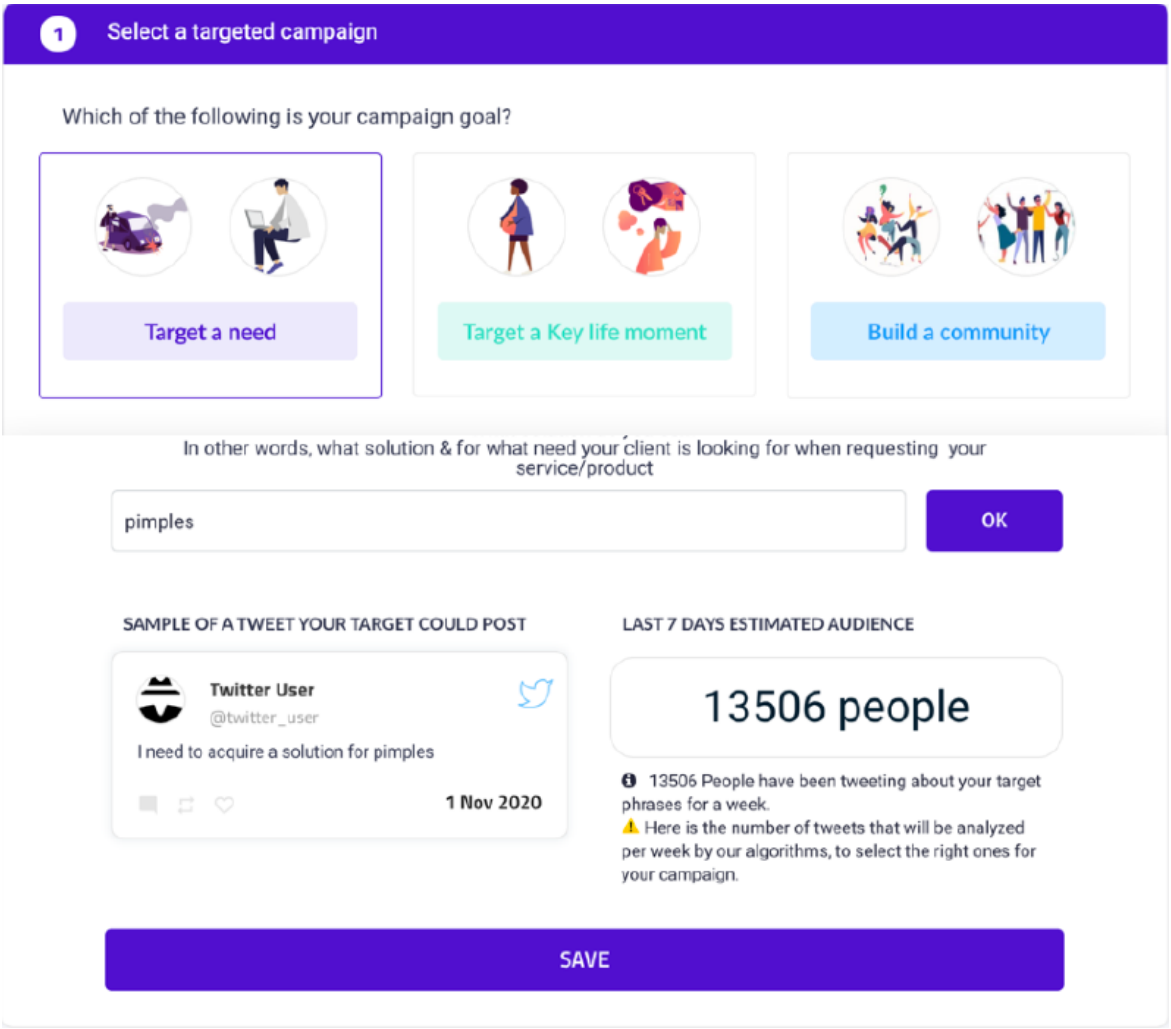


FIGURE 1.1: Préparation d'une campagne sur `app.makezu.io`

Une fois l'audience repérée, le client peut décider d'envoyer un message personnalisé à chaque personne. Ce message est soit sous la forme d'un tweet qui le mentionne, soit sous la forme d'un message privé. Le client choisit la mise en forme des messages envoyés à travers l'application Makezu. Une estimation de l'audience est visible une fois la campagne créée. De plus, l'engagement des utilisateurs sur Twitter (représenté par exemple par des clics sur un lien) est gardé en mémoire. Ainsi, le client peut avoir une idée de l'impact de ses campagnes et de l'engagement généré grâce à l'application Makezu. La figure 1.1 expose l'interface utilisateur de l'application permettant la création des campagnes. Bien que la création des campagnes soit gratuite, les opérations sur les campagnes sont limitées, nous allons voir quel est le modèle économique de Makezu dans la prochaine sous-section.

1.1.3 Modèle économique

L'application est disponible gratuitement grâce à une offre permettant d'effectuer jusqu'à dix opérations par semaine. Une opération est une activité sur Twitter qui s'adapte aux besoins du client et attire l'attention de son audience. Cela peut être un message privé envoyé automatiquement à

un nouveau follower ou le partage d'un article sur le compte Twitter du client (voir annexe 4.7). Un follower est une personne qui suit le compte Twitter du client. La fonctionnalité permettant le partage d'un article a été développée dans le cadre de ce stage (voir la sous-section 3.2.2). Des abonnements payants sont disponibles permettant d'effectuer plus d'opérations et d'avoir accès à des fonctionnalités exclusives. Par exemple, l'offre « Starter » propose aux clients d'effectuer jusqu'à 50 opérations par semaine à un coût de 29 dollars par mois. Avec l'offre payante, il est possible de suivre de près les followers acquis grâce aux campagnes, notamment en ayant accès à leurs informations publiques (voir annexe 4.8). Il est également possible d'avoir accès à de nouvelles automatisations comme l'envoi automatique d'un message à un nouveau follower. Nous verrons en quoi les fonctionnalités proposées par Makezu se distinguent de leurs concurrents, tout en restant dans une certaine tendance.

1.1.4 Concurrence et tendance

Makezu s'inscrit dans un marché qui se développe rapidement. Les principaux concurrents sont :

- **PhantomBuster** (`https://phantombuster.com`) qui permet l'extraction de données et l'automatisation d'envoi de messages sur une grande variété de plateforme dont LinkedIn, Github, Twitter, Facebook et Slack ;
- **Hootsuite** (`https://hootsuite.com/fr/`) qui est un outil de gestion de réseaux sociaux créé en 2008. Il prend la forme d'un tableau de bord et intègre les flux de différents réseaux sociaux comme Facebook, Twitter, LinkedIn et Wordpress. Il possède plusieurs millions d'inscrits ;
- **Sprinkrl** (`https://sprinklr.com/fr/`) qui a une grande capacité d'écoute sur plusieurs réseaux sociaux, canaux de messagerie, blogs, sites d'information et de discussions ainsi que des forums ;
- **Funnelai** (`https://funnelai.com`) qui analyse les écrits des utilisateurs de réseaux sociaux pour trouver de potentiels acheteurs. L'entreprise se concentre principalement sur l'industrie automobile ;
- **Twitter Ads** (`https://ads.twitter.com`) qui est le service concentré sur les publicités de Twitter. Il permet de sponsoriser des tweets et des comptes ainsi que de cibler des audiences très précisément grâce à leurs algorithmes internes.

Makezu est en compétition avec des entreprises anciennes qui connaissent leur secteur. Comme nous l'avons vu, Twitter est un concurrent direct de Makezu, ce qui peut poser de nombreux problèmes : si demain Twitter décide de révoquer l'accès à leur API, comment doit réagir Makezu ?

La nouveauté de Makezu est de proposer une relation-client plus personnalisée grâce à ses outils d'analyse de données et la possibilité d'avoir des relations *one-to-one*, ce qui est caractéristique d'une nouvelle forme de marketing ayant émergé dans les années 90, le « marketing personnalisé ». Cela correspond à un marketing individualisé, par opposition au marketing de masse. Récemment, la pratique du marketing individualisé a été amplifiée par les algorithmes d'apprentissage automatique qui permettent de cibler plus précisément les besoins fluctuants de chaque utilisateur. Après avoir présenté l'application Makezu et son insertion dans un marché compétitif, nous présenterons l'équipe qui se charge de son développement.

1.2 Présentation de l'équipe

Avec Hélène et Hugo, nous développons l'application Makezu. Hélène développe l'interface graphique grâce au constructeur d'application Bubble, un outil *no code* dont le fonctionnement sera plus détaillé dans la section 2.3. Elle se charge également de la relation avec les clients, et notamment de l'*onboarding* des bêta-testeurs, c'est à dire de la bonne intégration des futurs utilisateurs

au sein de l'application Makezu. En effet, pendant une grande partie de mon stage, Makezu reposait sur les retours des bêta-testeurs qui utilisaient gratuitement l'application. Les premiers clients ayant pris l'abonnement payant sont arrivés grâce au site AppSumo. Nous verrons plus en détail le fonctionnement de ce site dans la sous-section 2.4.3.

Hélène se charge de donner les principales directives. Les avis et les demandes des clients par rapport à Makezu sont recueillis. À partir de cela, nous réfléchissons tous ensemble à une manière de développer une solution qui pourrait satisfaire le client. Avec Hugo, je développe la solution technique tout en gardant le dialogue avec Hélène pour savoir si cela correspond aux besoins. Hugo se charge d'implémenter la solution finale sur le serveur. Une solution est acceptée lorsque nous sommes tous d'accord sur son bon fonctionnement et son utilité pour l'entreprise. Des correctifs peuvent être appliqués après sa mise en production en observant par exemple le registre des erreurs ou en prenant en compte les retours de nos utilisateurs. La section 2.3 présente plus en détail la façon dont l'application web Bubble et l'application hébergée sur le serveur communiquent.

Durant mon stage, j'étais invité à prendre des initiatives en termes de recommandations technologiques. Je pouvais participer au choix d'une architecture ainsi qu'à la *roadmap* future concernant le développement de nouvelles fonctionnalités. La *roadmap* est une feuille de route permettant de communiquer et partager une intention stratégique pour atteindre des objectifs. Je participais également à l'élaboration du cahier des charges. Les décisions étaient prises collectivement puisque chaque avis était pris en compte. Le chapitre suivant clarifie le sujet du stage, le planning, l'environnement technologique ainsi que la contribution apportée. Nous essayerons également de prendre un peu de recul sur le travail effectué.

Chapitre 2

Amélioration et évolution des algorithmes Makezu

2.1 Amélioration de l'application et évolution des algorithmes d'intelligence artificielle

Le sujet initial de mon stage est d'améliorer l'API de l'entreprise ainsi que de faire évoluer les algorithmes d'intelligence artificielle. Une API (*Application Programming Interface*) est une interface de programmation d'application. C'est un ensemble de classes, de méthodes et de fonctions qui sert de façade par laquelle un logiciel offre des services à d'autres logiciels.

Les algorithmes de Makezu ont pour objectif de détecter les besoins exprimés directement ou indirectement sur Twitter, les moments de vie clé et les centres d'intérêt des consommateurs qui s'expriment publiquement. Il est nécessaire d'élaborer des algorithmes permettant de traiter plusieurs langues, principalement l'anglais et le français.

Plus précisément, il s'agira de travailler sur :

— l'enrichissement des approches existantes en traitement du langage naturel comme le POS (*Part-Of-Speech tagging*) et le SD (*Syntactic Dependency Parsing*), ces deux approches seront expliquées dans la sous-section 3.2.1 ;
— l'amélioration de la précision des modèles, avec un seuil arbitraire fixé à au moins 75% ;
— le croisement des différentes approches pour un ciblage optimal de chaque besoin ;
— l'analyse d'un utilisateur Twitter par l'ensemble de ces tweets afin de créer un modèle de profilage ;
— le développement d'une fonctionnalité de réponse personnalisée qui dépendra du tweet d'origine ;
— la mise en place de processus de maintenance de l'API et des serveurs.

Cette liste est non-exhaustive et est plutôt présentée comme une ligne directrice. Durant mon stage, les directives ont sensiblement changé. Je me suis principalement intéressé à la création d'un modèle de profilage en utilisant plusieurs méthodes comme la classification multi-classes de données textuelles (voir le chapitre 4) ou la reconnaissance d'entités nommées (voir la sous-section 3.2.1). J'ai également explorer d'autres réseaux sociaux comme Quora ou LinkedIn (voir respectivement les sections 3.1 et 3.3) pour répondre au besoin d'automatisation des contenus de Makezu.

Même si la plupart de mes projets au sein de l'entreprise entrent en résonance avec des projets déjà existants, je n'ai pas repris ou continué le travail d'une personne. Comme indiqué dans la section 1.2, je ne suis pas un cahier de charges précis puisque je contribue activement à son élaboration.

Nous verrons dans la prochaine section quelles ont été plus précisément mes contributions au sein de l'entreprise.

2.2 Contributions

À mon arrivé, l'activité de Makezu était principalement concentrée sur la plateforme Twitter et sur la manière d'identifier les besoins et les moments de vie clés des utilisateurs. En développant des outils d'extraction de données concentrés sur les autres plateformes sociales (voir chapitre 3) comme Quora et LinkedIn, Makezu peut maintenant s'attaquer à l'automatisation des contenus sur celles-ci. De plus, l'entreprise possède de nouveaux terrains pour la recherche de prospects. Le profilage des utilisateurs est aussi plus précis grâce à la catégorisation du contenu textuel (voir chapitre 4) et l'analyse du genre (voir les sous-sections 3.2.4 et 3.2.5) qui permettront dans le futur d'établir des groupes d'utilisateurs ayant des propriétés communes. Les modèles de classification des textes pourront être utilisés sur du contenu textuel provenant des différentes plateformes et ne sont pas exclusifs à Twitter, ce qui permettra à Makezu d'avoir un large champ d'action. Ainsi, le défi futur est d'intégrer correctement ces nouveaux outils à l'application en respectant les contraintes computationnelles et les limites de requêtes imposées par les différentes plateformes. La mise en échelle de ces outils représente un sérieux enjeu. Dans la prochaine section, nous verrons plus en détail la nature de l'environnement technologique de Makezu pour avoir une vision globale du fonctionnement de l'application.

2.3 Environnement technologique

2.3.1 Bubble et AWS

Makezu est un ensemble de deux applications :

— la première application est codée entièrement en Python à l'aide du framework Flask [1]. Celle-ci se présente sous la forme d'une application web, qui peut être requêtée assez fréquemment par n'importe quelle plateforme. Il n'était pas nécessaire de développer un site web de bout en bout. Les interactions seront donc rapides et ponctuelles. L'application est hébergée sur une instance gratuite, fournit par *Amazon Web Services* (AWS) [2], qui est un serveur simple sans puissance de calcul graphique. Pour ce qui est du service d'enregistrement des données, on utilise un autre service disponible avec AWS, une base de données non-relationnelle DynamoDB ;

— la seconde est une application web développée grâce à l'outil Bubble [3], un constructeur d'application *no-code* ne demandant pas de lignes de code. Il n'est donc pas nécessaire d'avoir un développeur web. En effet, Bubble permet la génération d'une application web dynamique que l'on crée à l'aide d'une interface visuelle intuitive. C'est sur cette application que les utilisateurs naviguent et utilisent l'outil de Makezu. Elle permet la gestion des appels avec l'application hébergée sur AWS.

Makezu possède un site vitrine [4], qui expose notre application et permet de comprendre l'outil en lui-même, permettant aux visiteurs de s'inscrire sur la plateforme.

1. Une infrastructure logicielle open-source de développement web en Python.
2. Le service d'Amazon qui met à disposition des machines virtuelles permettant, par exemple, d'exécuter une application à distance.
3. `https://bubble.io/home`
4. `https://makezu.io`

2.3.2 Google Colab pour l'apprentissage automatique

L'apprentissage profond (*deep learning*) est un ensemble de méthodes d'apprentissage automatique qui tente de modéliser des données grâce à des transformations non linéaires. Un réseau de neurones est une méthode d'apprentissage profond. Étant donné que nous n'avons pas à disposition des ordinateurs avec GPU pour entraîner nos modèles d'apprentissage profond, nous utilisons Colaboratory [1] qui est un produit de Google Research. Colab permet à n'importe qui d'écrire et d'exécuter du code Python par le biais du navigateur. C'est un environnement particulièrement adapté à l'analyse de données et à l'éducation. En termes plus techniques, Colab est un service hébergé de *notebooks* Jupyter [2] qui ne nécessite aucune configuration et permet d'accéder gratuitement à des ressources informatiques comme des CPU (*Central Processing Unit*), des GPU (*Graphics Processing Unit*) et maintenant même des TPU (*Tensor Processing Unit*). Un TPU est un type de processeur dédié au calcul d'apprentissage des réseaux de neurones développé par Google. Un modèle de classification est entraîné avec un TPU dans la section 4.2. Colab nous permet principalement d'entraîner nos modèles d'apprentissage profond et d'effectuer des points de sauvegarde en liant l'environnement Colab à un environnement de stockage Google Drive. Le service est gratuit mais possède des contraintes. Colab peut fournir un accès gratuit aux ressources grâce à un système de limites d'utilisation dynamiques. Celles-ci fluctuent et n'offrent ni garantie d'accès, ni ressources illimitées. Les machines virtuelles sont soumises à une durée d'expiration d'environ 9 heures, ce qui est assez peu si nous voulons entraîner nos modèles entièrement et effectuer différents tests. Dans la prochain section, nous présenterons quelques étapes importantes durant la période de stage.

2.4 Planning

2.4.1 Incubateur Station F

Station F est l'un des plus grands campus de startups au monde, inauguré en 2017, réparti sur 34 000 mètres carrés. Initialement, le stage aurait dû se passer dans ces locaux, dans le treizième arrondissement de Paris. Ce lieu a changé trois semaines avant le début de mon stage. Celui-ci s'est donc passé dans l'incubateur de startups WILLA situé dans le deuxième arrondissement. Cela devait être temporaire mais la date de notre arrivée à Station F n'a fait que reculer. Passant de septembre à octobre, puis janvier, et finalement cela ne s'est jamais fait. Malheureusement, WILLA est très loin de mon logement comparé à Station F.

2.4.2 L'Oréal et Founders Factory

Fin octobre, nous avons été accepté chez Founders Factory, un accélérateur et incubateur Londonien dans le digital en partenariat avec L'Oréal. L'entreprise met à disposition de chaque startup une équipe interne d'experts qui leur donne un appui opérationnel et tient un rôle de « mentor » en plus d'une aide financière. Elle participe chaque année à la croissance d'une dizaine de startups en démarrage. La durée du programme est de six mois. Une équipe technique est à notre disposition, nous pouvons par exemple demander de l'aide au *Data Scientist* de Founders Factory et avoir des conseils sur l'architecture ou les différentes approches possibles pour résoudre un problème. En échange de ces aides, l'Oréal espère obtenir un « retour sur investissement ». En effet, certaines fonctionnalités de Makezu intéressent l'entreprise, notamment la possibilité d'envoyer des messages personnalisés sur Twitter mais aussi sur d'autres plateformes. L'Oréal utilise déjà un de nos concurrents, mentionné dans la sous-section 1.1.4, Sprinkrl. Néanmoins, l'entreprise aimerait coupler cette

1. `https://colab.research.google.com`
2. Jupyter permet de réaliser des *notebooks*, c'est-à-dire des programmes contenant à la fois du texte en *markdown* et du code en Python. Cela est très utile pour exécuter certains bouts de code et non l'ensemble.

solution avec la notre pour ainsi synergiser les qualités : avoir la capacité d'écoute puissante de Sprinkrl couplée avec les automatisations personnalisées de Makezu.

Une réunion est effectuée chaque mois avec une quinzaine de personne où nous faisons état de l'avancée et de ce qu'il reste à faire avec l'équipe. Un environnement Slack est également dédié à la communication entre l'incubateur et notre entreprise.

2.4.3 Lancement sur AppSumo

AppSumo est un site web d'offres quotidiennes pour les produits distribués numériquement et les services en ligne. Makezu passe par ce service pour se faire connaître et avoir des clients rapidement [1]. En effet, le site attire énormément de clients potentiels puisqu'il propose des forfaits peu onéreux. Lors de notre lancement sur cette plateforme, début décembre, le forfait pour l'abonnement à vie était de 69 dollars contre 420 dollars normalement. Ce lancement a permis à Makezu d'avoir sa première rentrée d'argent. Également, nous avons pu récolter divers avis des clients. Une des remarques pertinentes était le besoin d'utiliser les services d'automatisation de Makezu pour d'autres plateformes sociales comme LinkedIn. Nous terminons ce chapitre par une prise de recul sur le travail effectué durant ce stage.

2.5 Prise de recul

L'intérêt de notre travail pour l'entreprise a été d'améliorer l'application existante, en ajoutant notamment des fonctionnalités d'automatisation, mais aussi d'élargir le champ d'action de l'application en élaborant des programmes de récupération de données sur d'autres plateformes sociales que Twitter.

Certains programmes ont besoin d'une maintenance régulière dus à leur nature relationnelle. Par exemple, notre extracteur d'information sur Quora, exposé dans la section 3.1, utilise l'interface du site web pour son bon fonctionnement. Si l'interface vient à changer, il faudra modifier une partie du code pour prendre en compte ce changement. Les programmes d'extraction de données utilisant le navigateur sont consubstantielles des sites web qui agissent comme support d'information. Un programme n'est jamais complètement terminé s'il n'est pas fermé sur lui-même. Il convient également de continuer à améliorer les modèles d'analyse de données. De nouveaux ensembles de données peuvent être disponibles et représenter un apport d'information significatif. L'amélioration d'un modèle passe par l'enrichissement des données, mais également par le choix de l'architecture. Il faudra être au courant des dernières avancés dans le domaine du traitement du langage naturel, et ne pas hésiter à combiner diverses approches.

Pour finir, il me semble important d'entretenir une discussion sur l'éthique des modèles de traitement de données. L'automatisation du contenu peut représenter un danger pour la société. Le client peut avoir l'impression qu'il « commande » l'automatisation, en proposant par exemple un sujet qui l'intéresse. Néanmoins, il est important de rappeler qu'un algorithme d'automatisation ne créé pas, il ne fait que reproduire quelque chose qui existait déjà. Le problème réside dans la sélection du contenu. L'algorithme effectue des choix qui sont implicitement supportés par des valeurs. Par exemple, le programme qui permet de partager des articles d'actualités sur Twitter, dont l'élaboration est exposée dans la sous-section 3.2.2, ne peut exister sans l'algorithme interne de Google Actualités qui présente, et donc sélectionne, les articles en amont. Cette opacité technique peut représenter un danger puisque, sans le savoir, nous pouvons véhiculer des valeurs qui ne sont pas les nôtres. Par transitivité, le client pourrait véhiculer des valeurs qui sont contraires à son image, ce qui pourrait être très contre-productif. Le client a la possibilité de bannir certains sites pour ne pas que ces derniers soient mis en avant, mais cela ne résout que partiellement le

1. `https://appsumo.com/makezu/`

problème. En conclusion, il est important d'avoir en tête le caractère non-neutre de la technique lors de l'élaboration d'un algorithme. Nous aurons l'occasion de rediscuter de ce point dans la conclusion de ce rapport. Après avoir clarifier le sujet du stage, l'environnement technologique, le planning et la contribution apportée, nous allons décrire mes réalisations en commençant par les outils d'extraction et d'automatisation du contenu sur les différentes plateformes sociales.

Chapitre 3

Extraction et automatisation du contenu sur les différentes plateformes sociales

3.1 Quora

3.1.1 Présentation de Quora et des outils utilisés

Quora est un réseau social qui permet à ses utilisateurs de créer, d'éditer et d'organiser des questions-réponses. Les questions-réponses sont organisées par sujets. Il est assez difficile de travailler sur Quora puisque ce dernier ne possède pas d'API. De plus, les applications disponibles sur Github [2] qui interagissent avec Quora sont limitées ou obsolètes. Il est donc nécessaire de créer un programme « de zero ». Cela demande du temps et une maîtrise de technologies nouvelles, néanmoins, nous pourrons proposer des services qui ne sont pas encore disponibles sur le marché. On utilise deux librairies Python très intéressantes :

— **selenium**, qui est un projet englobant un ensemble d'outil et de librairies rendant possible l'automatisation de navigateur web. Grâce à l'écriture de script, nous pouvons réaliser automatiquement des actions dans un navigateur comme la visite d'une page, le clic sur un lien et le remplissage d'un formulaire. Il est également possible de récupérer les résultats de ces actions. Ainsi, il est possible de simuler la navigation d'un utilisateur sur Quora avec son navigateur préféré. Le but est de récupérer les informations que l'utilisateur voit à l'écran lorsqu'il navigue sur une page ;

— **beautiful soup**, qui est une bibliothèque Python d'analyse syntaxique de documents HTML et XML. Elle produit un arbre syntaxique qui peut être utilisé pour chercher des éléments ou les modifier. Il est ainsi possible de chercher le contenu d'une balise HTML spécifique. Nous utilisons cette bibliothèque pour extraire le code HTML d'une page atteinte grâce à Selenium.

3.1.2 Méthodologie et fonctionnalités principales

Une fois les technologies maîtrisées, la récupération de n'importe quelle information disponible à l'écran d'un utilisateur lorsqu'il navigue sur une page Quora est aisée. Voici le procédé utilisé sur un exemple simple, l'extraction d'une question Quora :

2. Un service web d'hébergement et de gestion de développement de logiciels.

— nous ouvrons manuellement la page contenant la question Quora en navigation privé. Étant donné que nous n'utilisons pas de cookies lors de nos sessions Selenium, il est nécessaire de faire de même avec notre navigateur pour être sûr de pouvoir récupérer les mêmes informations affichées ;
— nous récupérons avec Beautiful Soup le contenu HTML de la page. Nous cherchons où se trouve la question avec une recherche de terme, un simple raccourci clavier est suffisant. Le but est de retrouver la question affichée à l'écran dans le code HTML récupéré ;
— une fois le contenu de la question trouvé dans la réponse HTML de Beautiful Soup, nous pouvons remonter les balises pour avoir notre chemin. Avec ce procédé, nous pouvons remonter vers la balise qui englobe toutes les questions de la page. Il suffit alors d'extraire le texte de ces balises.

Nous avons incorporé à l'aide de ce procédé les fonctionnalités suivantes à notre programme :

— la récupération des questions et des réponses d'un sujet Quora. Par exemple, pour le sujet *Computer Science*, le navigateur de Selenium se connecte à la page `https://www.quora.com/topic/Computer-Science` et extrait les informations visibles en simulant le défilement de la page. Seulement les questions mises en avant par l'algorithme interne de Quora sont récupérés. Ainsi, nous n'avons pas accès à toutes les questions/réponses d'un sujet ;
— la récupération des sujets similaires. Par exemple, parmi les informations extraites sur le sujet *Computer Science*, nous pouvons récupérer *Algorithms*, *Machine Learning*, *Computers*, *Technology*, etc. Cela nous permet d'élaborer des *clusters*[1] de sujets. En effet, nous pouvons récursivement récupérer les sujets similaires pour avoir des groupes de sujets. Cela est possible puisque les sujets Quora forment des graphes connexes. Autrement, si les graphes n'étaient pas connexes, alors cela voudrait dire qu'à partir de n'importe quel sujet on pourrait atteindre un autre sujet ;
— la récupération du nombre de personnes qui suivent un sujet. Nous pouvons ainsi filtrer les sujets Quora selon le nombre de suiveurs ;
— la récupération des questions similaires à une question. Ceci est le seul moyen pour récupérer des questions qui n'ont pas été mises en avant par l'algorithme de Quora sur la page d'un sujet ;
— le profilage d'une personne. Selenium se connecte sur le profil d'une personne et extrait le plus d'information possible comme la biographie, le nombre de suiveurs, le nombre de vues sur les réponses, la localisation, les études, le métier actuel et les sujets préférés.

Étant donné que l'interface d'un site web n'est jamais fixe, et que notre approche se concentre principalement sur l'architecture visible de Quora, il est fort probable que celle-ci change dans le temps. il faudra donc construire des correctifs et changer les éléments « obsolètes » pour les remplacer par les nouveaux éléments utilisés. Si Quora décide de refaire intégralement l'interface de son site, il faudra reconstruire le programme.

Il est important de préciser qu'il est interdit, selon les règles fixées par Quora, de créer du contenu « dérivé » à partir des données recueillies. Il n'est donc pas possible d'utiliser des algorithmes d'apprentissage automatique.

3.1.3 Possibles améliorations du programme de récupération des informations

Il serait intéressant de pouvoir récupérer plusieurs questions et réponses d'un seul utilisateur pour pouvoir améliorer la fonctionnalité de profilage. Nous pouvons également croiser les informations de Quora avec celles d'autres plateformes en cherchant par exemple si un utilisateur existe sur Twitter et Quora en même temps. Dans la sous-section 3.2.3, nous allons utiliser Quora pour

1. Anglicisme qui signifie un groupement.

automatiser le partage de contenu sur Twitter. Nous avons tourné le programme quelques jours sur l'ordinateur personnel. Plus de 3000 sujets et 30 000 questions et réponses ont été récupérés avec les utilisateurs correspondant. Il serait plus efficace d'exécuter le programme sur un serveur et ainsi récupérer des informations sans interruption pour alimenter continuellement la base de données des questions/réponses et sujets. Actuellement, le client doit écrire parfaitement le nom d'un sujet pour que le programme ouvre le fichier CSV correspondant. Un fichier CSV (*Comma-separated values*) est un format texte représentant des données tabulaires sous forme de valeurs séparées par des virgules. Il serait intéressant de calculer un indice de similarité sémantique entre ce qu'inscrit le client et les sujets Quora stockés pour lui proposer des sujets similaires à ses envies. Une autre méthode est de chercher le sujet du client sur Quora et de récupérer les sujets retournées par la plateforme. Il suffit ensuite de faire appel à l'algorithme qui récupère les questions et réponses d'un sujet précis. Dans la prochaine section, nous présenterons les nouvelles fonctionnalités de l'application Makezu pour la plateforme Twitter.

3.2 Twitter

3.2.1 Analyse des tendances

Sur Twitter, une tendance est un sujet populaire qui apparaît à un instant t et qui est déterminé par un algorithme interne de Twitter selon la localisation mais aussi les centres d'intérêt et les abonnements d'un utilisateur. Makezu n'exploitait pas encore cette fonctionnalité. Celle-ci peut-être intéressante pour recommander des tendances ayant des points communs avec le domaine d'un client. Par exemple, si un client est un vendeur de voiture et qu'un nouveau modèle de voiture entre en tendance, il serait intéressant de lui proposer cette tendance et d'analyser les utilisateurs qui l'animent.

Il existe deux types de tendance. Des *hashtag* comme « #CouvreFeu18h » peuvent apparaître en tendance si ceux-ci sont utilisés massivement. Souvent, les *hashtag* sont utilisés à la fin d'une phrase pour ajouter du sens. Il existe aussi tout simplement des tendances représentées par un mot ou un groupe de mot comme « Jack Lang ». Ce dernier type de tendance est intéressant puisqu'il est généralement utilisé à l'intérieur d'une phrase, il est ainsi insérer directement dans un contexte précis. Nous aimerions, en analysant les tweets sous une tendance, déterminer les mots en rapport avec cette tendance pour ainsi ajouter de la valeur à celle-ci. Il sera ainsi plus facile de suggérer une tendance pour un client si nous avons plusieurs mots qui informent du contexte dans lequel cette tendance apparaît.

Nous communiquons avec l'API de Twitter grâce à la librairie Python **Tweepy**. C'est à travers cette librairie que nous pouvons envoyer ou récupérer des tweets grâce à la clé API d'un compte développeur Twitter. Tous les travaux portant sur Twitter qui ont été effectués durant mon stage utilise cette librairie. Nous utilisons Tweepy pour récupérer les identifiants uniques de chaque pays. Les identifiants uniques sont des *Where On Earth IDentifier* (WOEID). Un WOEID est un identifiant de référence unique pour chaque élément ayant une localisation. Il existe également des identifiants uniques pour des pairs de pays/ville. Au total, nous pouvons récupérer les identifiants de 60 pays et plus de 400 pairs de pays/ville. Les WOEID sont utilisés pour récupérer les tendances d'un pays donné à travers une requête Tweepy.

Une cinquantaine de tendances pour un pays est récupérée. Pour chaque tendance, des centaines de tweets sont récupérées. Les entités de chaque tweet sont extraites. Celles qui reviennent le plus souvent parmi la totalité des tweets sont retournées et permettent - normalement - de renseigner sur le contexte d'une tendance. L'analyse s'effectue grâce à deux techniques de traitement du langage naturel : le *Part-Of-Speech tagging* (POS) ou étiquetage morpho-syntaxique en français, et le *Name Entity Recognition* ou reconnaissance des entitées nommées. L'étiquetage morpho-syntaxique est un processus qui consiste à associer aux mots d'un texte les informations grammaticales correspondantes comme la partie du discours, le genre, le nombre, etc. Un exemple est illustré sur la figure 3.1 [1].

À partir de l'étiquetage, les entités sont déterminées. Par exemple, pour la phrase *Manchester United is looking to sign Harry Kane for $90 million*, les entités « Manchester United » (une organisation), « Harry Kane » (une personnalité), « $90 *million* » (un nombre) sont retournées. Nous utilisons la librairie **SpaCy** pour ces exemples, celle-ci est entièrement utilisée pour notre projet d'analyse des tendances. Cette librairie Python est construite pour effectuer du traitement du langage naturel rapidement, ce qui est parfait pour mettre en production des outils d'analyse sans construire des modèles à partir zéro. En effet, SpaCy incorpore des modèles pré-entraînés de réseaux de neurones atteignant l'état de l'art et permettant le POS *tagging* et la reconnaissance des entités nommées. Il supporte également la tokenisation dans une soixantaine de langues différentes. La tokenisation est le fait de segmenter une phrase en plusieurs mots. Chaque mot est ainsi un

1. `https://stackabuse.com/python-for-nlp-parts-of-speech-tagging-and-named-entity-recognition/`

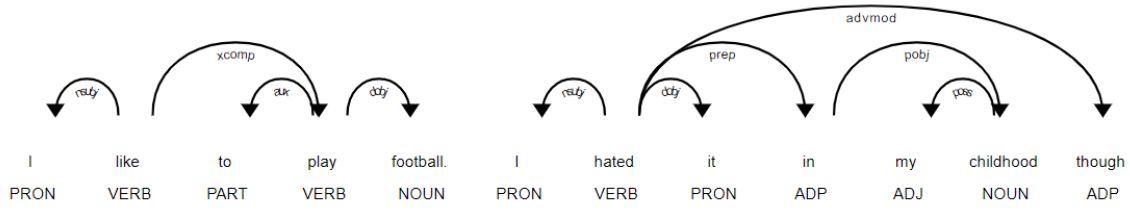


FIGURE 3.1: POS tagging pour la phrase : *I like to play football. I hated it in my childhood though*

token qui pourra être encoder sous la forme d'un vecteur pour pouvoir être utilisé dans un modèle d'apprentissage profond. Le représentation d'un token sous la forme d'un vecteur est un problème d'optimisation. En effet, il est nécessaire d'apprendre à un modèle à encoder l'information latente des mots. SpaCy permet la vectorisation des mots grâce à un modèle entraîné sur Common Crawl, *Global Vectors for Word Representation*[1] (GloVe) [1]. SpaCy basé sur GloVe contient 1.1 millions de vecteurs uniques de dimension 300. Ainsi, un vecteur unique est attribué pour chaque token s'il existe dans la liste. Pour représenter une phrase, comme un tweet, il suffit d'agréger les différents vecteurs pour avoir un vecteur qui embrasse une représentation globale. Stanford, à l'origine de GloVe, met aussi à disposition 1.2 millions de vecteurs uniques pré-entraînés sur 2 milliards de tweet disponible en 25, 50, 100 et 200 dimensions.

Les représentations vectorielles des mots ouvrent la voie à de nombreuses applications. Nous pouvons faire de la « recherche des plus proches voisins » (*nearest neighbors*) en calculant la distance euclidienne ou la similarité cosinus entre deux vecteurs pour ainsi avoir une idée de la similarité sémantique de ces mots. Les structures latentes peuvent également être représentées. La différence entre deux vecteurs permet de capturer le sens de la juxtaposition des deux mots. Sur la figure 3.2, nous voyons que le concept sous-jacent qui distingue sémantiquement l'homme de la femme (comme le sexe ou le genre) est similaire à d'autres paires de mots comme avec reine et roi ou frère et soeur.

Le procédé est simple. Des centaines de tweet sont récupérées sous une tendance. Chaque tweet est découpé en plusieurs tokens. Les tokens sont représentés sous forme de vecteurs. À partir des vecteurs, des modèles d'apprentissage profond pré-entraînés sont utilisés pour l'étiquetage morpho-syntaxique et la reconnaissance des entités nommées. Les entités de chaque tweet sont stockés et ceux qui apparaissent le plus souvent parmi la totalité des tweets récupérés sont retournés. Par exemple, pour la tendance « Macron » , les entités « République », « France » et « Panthéon » sont retournées. L'entité « Panthéon » est intéressante, en regardant les tweets sous la tendance nous observons que Macron venait de faire un discours au Panthéon, ce qui a suscité des réactions sur Twitter.

Dans le chapitre 4, nous verrons une autre manière d'analyser les tendances qui se couple très bien avec l'analyse des entités. En effet, le contexte peut rester flou même si nous relions le mot en tendance avec ses entités. Nous proposons de faire de la classification du contenu textuel pour déterminer la catégorie de chaque tendance. Dans la prochaine sous-section, nous présentons la fonctionnalité de partage d'articles sur Twitter.

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

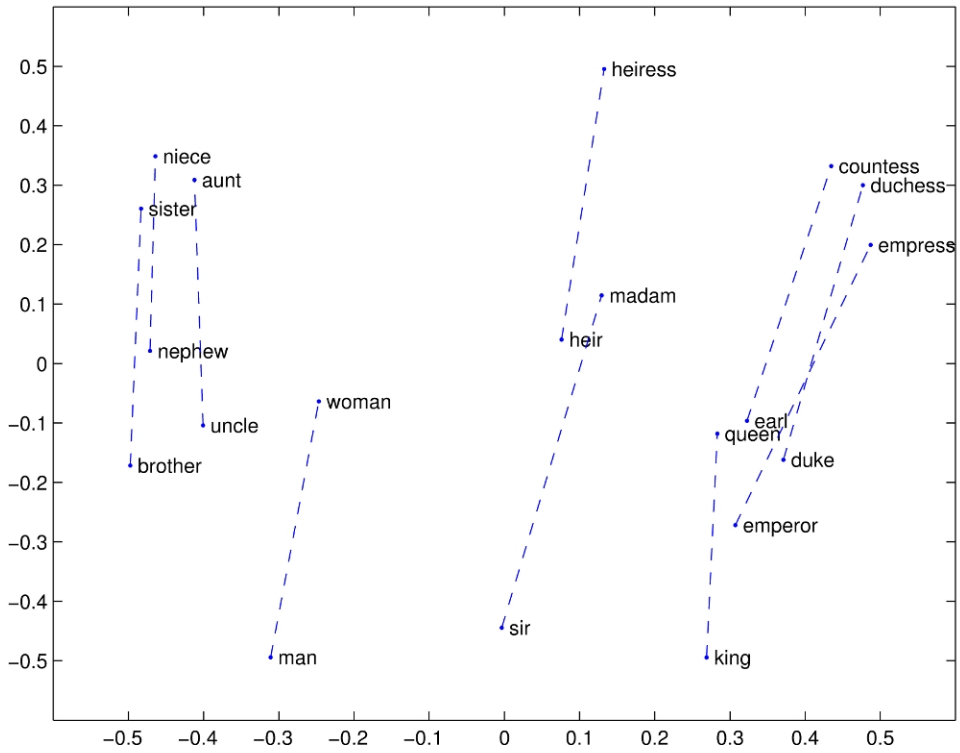


FIGURE 3.2: Représentation des structures latentes avec GloVe

3.2.2 Partage d'articles sur Twitter

Une des demandes de Makezu est l'automatisation des comptes Twitter de ses clients, et notamment l'enrichissement du contenu partagé. Nous allons créer un programme qui permet de partager des articles automatiquement selon des sujets que le client aura renseigné. Nous utilisons le flux RSS [1] de Google Actualités. C'est un service en ligne gratuit développé par Google qui présente de façon automatisée des articles d'information en provenance de sources sur le web. Il fonctionne de la même manière qu'un moteur de recherche, en n'indexant que les articles de presse. Le programme effectue les étapes suivantes :

- la construction à partir du sujet du client d'une requête GET qui sera adressée à Google actualité. Par exemple, si nous voulons récupéré des articles qui parlent du bitcoin, nous construisons cette URL : `https://news.google.com/rss/search?q=bitcoin`. En réalité, l'URL est plus complexe puisqu'il y a également possibilité de sélectionner la langue des articles ainsi qu'une période de temps (article de la semaine, du mois, de l'année etc.) ;
- l'envoi de la requête et la récupération du contenu de la réponse pour extraire les liens des articles, à l'aide de la librairie Python **requests** [2]. C'est une librairie HTTP permettant d'effectuer des requêtes GET, POST, UPDATE et DELETE ;
- l'extraction des données importantes d'un article comme son titre et sa description, à l'aide de la librairie Python **news-please** [3]. Cette librairie extrait les métadonnées des articles en analysant les balises HTML ;

1. Un flux RSS est un format de fichier particulier dont le contenu est produit automatiquement en fonction des mises à jour d'un site web.
2. `https://fr.python-requests.org/en/latest/`
3. `https://github.com/fhamborg/news-please`

— l'envoi d'un tweet composé du titre, de la description et du sujet. Un exemple est illustré sur la figure 3.3.

FIGURE 3.3: Partage automatique d'articles sur le sujet « bitcoin »

3.2.3 Partage des questions et réponses Quora

Toujours dans le but d'enrichir le contenu partagé sur Twitter, nous pouvons utiliser ce qui a été préalablement développé sur Quora dans la section 3.1. Le but est d'utiliser les discussions ou *thread* Twitter en liaison avec Quora. Un *thread* est une série de tweets connectés entre eux et publiés par une personne. La discussion permet de fournir plus de contexte et de donner plus d'informations en reliant plusieurs tweets. Quora est une opportunité pour Makezu d'exploiter pour la première fois cette fonctionnalité.

Nous reprenons sensiblement le même schéma que pour le partage d'article présenté dans la sous-section précédente :

— le client choisit un sujet présent sur Quora qui l'intéresse. Le programme pioche aléatoirement dans les questions relatives au sujet. Ces questions ont été préalablement récupérées et sont stockées dans un fichier ;
— le contenu de la question est utilisé pour effectuer le premier tweet de la discussion. Le client peut ajouter du texte additionnel et le sujet Quora est présent sous la forme d'un *hashtag* ;
— une fois la question postée, le programme présenté dans la section 3.1 récupère à l'aide des librairies Python Selenium et Beautiful Soup les réponses. Quelques réponses sont postées selon le choix du client. La dernière réponse fait référence au lien Quora de la question.

Il est nécessaire de bien optimiser l'espace et de pouvoir constamment exploiter la limite des 280 caractères sur les tweets. Nous espérons que cette fonctionnalité permettra aux clients de susciter plus d'engagement avec leurs communautés sur des sujets communs. Vous trouverez ci-dessous,

sur la figure 3.4, un exemple de discussion générée automatiquement par le programme. Dans la prochaine section, nous allons analyser les utilisateurs pour déterminer leur genre.

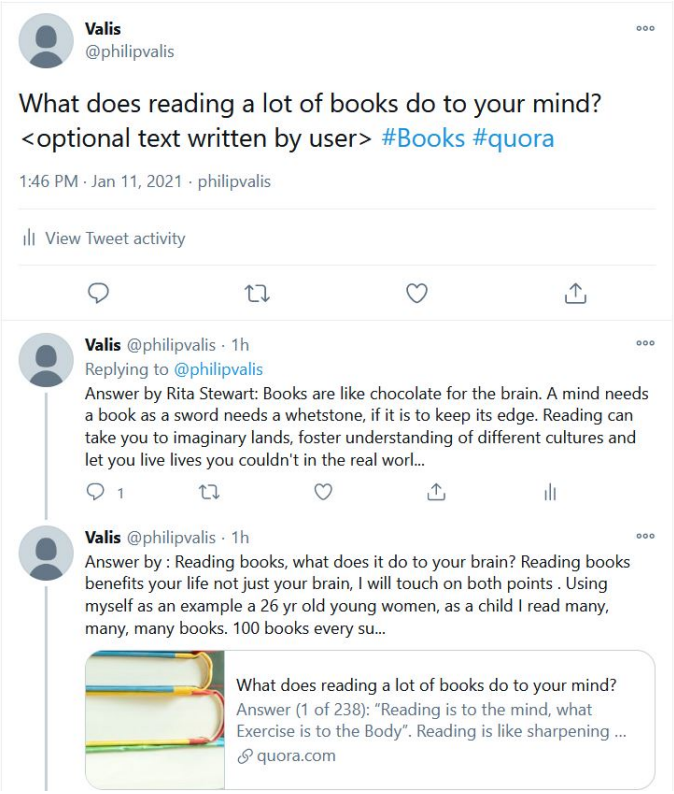


FIGURE 3.4: Exemple de discussion autour du sujet Quora *Books*

3.2.4 Analyse du nom d'utilisateur pour déterminer le genre

Certains clients aimeraient pouvoir cibler les utilisateurs selon leur genre. Par exemple, nous avons un client qui était intéressé par le fait de recruter des femmes dans le domaine des nouvelles technologies. Étant donné qu'il n'est pas possible d'avoir des informations explicites sur le genre d'une personne à travers son profil Twitter, nous devons récolter des informations implicites qui pourraient nous aider. Il est important de préciser que l'identité numérique ne reflète pas toujours l'identité « réelle ». Une identité numérique n'a pas forcément un genre correspondant. Il existera toujours un flou dans la détermination du genre d'une personne à travers ses traces numériques.

Nous proposons d'analyser le nom qu'utilise un utilisateur sur Twitter pour déterminer son genre. On utilise *World Gender Name Dictionary* [2], un dictionnaire qui inclut 6.2 millions de noms provenant de 182 pays différents pour distinguer le genre. Notre programme découpe le nom d'utilisateur en plusieurs morceaux et vérifie si un des morceaux se trouve dans la base de donnée de nom. Si oui, alors le genre correspondant est retourné. Un problème réside dans le découpage des pseudonymes. L'efficacité de la méthode dépend grandement de la manière dont on découpe le pseudonyme d'une personne. Par exemple, @SteveStuWill donnera « Steve Stu Will » avec un découpage par les majuscules. « Steve » se trouve dans le dictionnaire donc le genre correspondant est retourné. Qu'en est-il de @lexfridman ou de @nietzsche123 ? Comment effectuer le bon découpage ? La performance de cette approche dépend fortement de la manière dont on découpe

les noms. Ainsi, elle peut être améliorée avec de nouvelles techniques de découpage. Nous pouvons également chercher les informations autre part, nous allons maintenant analyser la biographie des comptes Twitter.

3.2.5 Analyse de la biographie pour déterminer le genre

Un utilisateur peut renseigner en quelques caractères des informations sur lui-même à travers sa biographie publique. Nous utilisons regex pour traiter les biographies et chercher des occurrences intéressantes. De plus en plus de personnes sur Twitter indiquent dans leur biographie leur genre. Par exemple, il n'est pas rare de trouver l'occurrence de *he/him* ou *she/her*. De même, et surtout chez les américains, il est courant de préciser sa place dans la famille en se définissant comme *mother*, *husband*, *dad* etc. Il convient alors d'établir de simples règles regex pour traiter les biographies des utilisateurs. Il est important de préciser que l'algorithme incorpore une vision « hétéronormale » du genre. Par exemple, si une personne parle de sa femme en écrivant *my wife* dans sa biographie, cela ne signifie pas forcément que la personne est un homme. Ainsi, nous pensons qu'il faudrait utiliser seulement les informations explicites comme *he/him*, néanmoins cela concerne moins de 5% des utilisateurs [1]. Il serait intéressant de coupler cette méthode avec une analyse des photos de profil. Actuellement, les réseaux de neurones convolutifs utilisés en vision par ordinateur réussissent très bien ce genre de tâche. Nous pouvons aussi analyser les tweets pour déterminer le genre si le langage le permet. Par exemple, « je suis énervée » est accordé au féminin donc nous pouvons déduire que la personne qui écrit cela est une femme.

FIGURE 3.5: Profil d'un utilisateur Twitter accompagné de sa biographie

Nous avons pu augmenter les fonctionnalités de l'application Makezu sur Twitter en analysant les tendances à l'aide de la reconnaissance des entités nommées. Le partage de contenu est aussi rendu possible grâce aux outils d'extraction de données provenant de Quora et de Google Actualités. La prochaine section présente l'automatisation du contenu sur la plateforme LinkedIn.

1. Selon nos observations sur des milliers d'utilisateurs.

3.3 LinkedIn

LinkedIn est un réseau social professionnel créé en 2002. Ce réseau revendique plusieurs centaines de millions d'utilisateurs actifs par mois. LinkedIn est un nouveau terrain pour Makezu. Le but est d'explorer les possibilités de cette plateforme. Après avoir créé une application développeur sur le site LinkedIn, nous pouvons avoir accès à l'API *Share*. Cette API permet de partager du texte sur le fil d'actualité LinkedIn, et ainsi partager du contenu à son réseau professionnel. L'intérêt pour Makezu est de pouvoir automatiser le partage de contenu de ses clients sur LinkedIn.

Il n'existe pas d'API officiel disponible en Python. De plus, les requêtes proposées sur Postman [1], un outil permettant de manipuler facilement les différentes API, sont assez limités et ne permettent pas le partage de texte. À l'aide de la documentation LinkedIn [2] et de la librairie requests, nous allons créer une application de partage de texte sur le fil d'actualité. Cette dernière librairie nous permettra de communiquer avec l'application LinkedIn.

Nous utilisons des requêtes authentifiées avec la clé fournie par LinkedIn. Le schéma de notre programme est le suivant :

— le programme récupère l'identifiant du compte qui a renseigné une clé valide. Si le compte n'a pas accepté l'application Makezu, alors rien ne se passe ;
— le client renseigne les informations qu'il veut partager, comme le texte et le lien du partage. Les informations sont encapsulées dans un fichier *JavaScript Object Notation* (JSON). Ce type de fichier permet de représenter de l'information structurée. Nous utilisons JSON pour envoyer des données à travers des requêtes HTTP ;
— la requête POST contenant les données est envoyée. Nous récupérons le code réponse de la requête. Si celui-ci est égale à 201, alors la requête a été correctement exécutée et le contenu est partagé sur le fil d'actualité du client.

Le même procédé est utilisé pour éditer, sauvegarder ou supprimer du contenu partagé. Pour supprimer du contenu, nous utiliserons une requête DELETE. Un exemple du contenu partagé est illustré sur la figure 3.6.

Après avoir exposé les outils construits pour extraire les données des différents réseaux sociaux, dans le prochaine chapitre, des modèles de classification sont élaborés pour enrichir l'analyse des tendances et des utilisateurs.

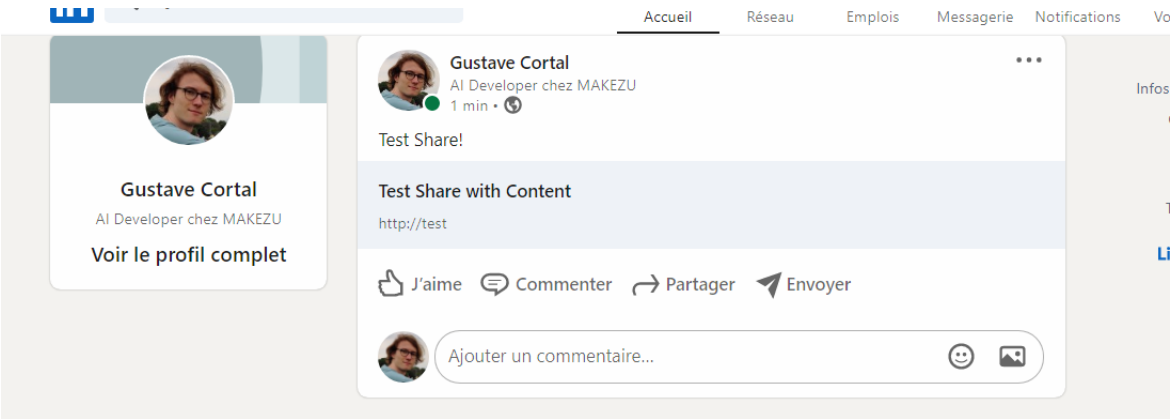


FIGURE 3.6: Exemple de contenu partagé sur le fil d'actualité LinkedIn

1. `https://www.postman.com`
2. `https://www.linkedin.com/developers/`

Chapitre 4

Catégorisation des textes pour l'analyse des tendances et le profilage des utilisateurs sur Twitter

4.1 Introduction

4.1.1 Intérêts pour Makezu

Makezu a besoin d'analyser les tendances sur Twitter et de les comprendre. Notamment, nous aimerions catégoriser celles-ci. Par exemple, si « Macron » entre en tendance Twitter, il serait intéressant de pouvoir exprimer la catégorie de cette tendance, qui est « politique ». Cela permettra à Makezu de proposer aux clients des tendances qui pourraient l'intéresser, mais aussi de cibler les personnes qui parlent sur ces tendances. Nous proposons de travailler sur un tweet individuellement pour pouvoir déterminer sa catégorie. La démarche sera de collecter assez de tweets sous une certaine tendance, de prédire les catégories respectives et de retourner la moyenne de ces catégories. Ainsi, on pourra par exemple récolter 300 tweets et caractériser la tendance « Macron » comme appartenant à la classe « politique » à 82% et comme appartenant à la classe « économie » à 18%. Nous pouvons utiliser une démarche similaire pour l'analyse de tweets des utilisateurs et ainsi établir des profiles. Au lieu de récolter des tweets sous certaines tendances, nous pouvons par exemple récolter les 100 derniers tweets d'un utilisateur pour avoir une idée de ses centres d'intérêt. Nous pouvons alors élaborer une liste des utilisateurs selon leurs centres d'intérêt et mettre cela à disposition pour nos clients qui voudraient cibler une certaine audience. Par exemple, si un de nos clients est un vendeur de ballons de football, nous pouvons lui proposer une liste d'utilisateurs intéressés par le sport. Les prochaines sections présentent l'élaboration de deux modèles d'apprentissage profond permettant la classification de texte, en anglais et en français [2].

4.1.2 Éléments d'histoire des modèles profonds de langage

Un grand changement dans le traitement du langage naturel a été l'introduction du pré-entraînement non-supervisé des représentations du langage en utilisant seulement du texte en clair, c'est-à-dire

2. Je suis en autonomie sur ce projet du début jusqu'à la fin, que cela soit sur le choix des données, l'architecture des modèles ou l'approche générale du problème.

du texte non-annoté et non modifié [1] depuis la source. Avant, les méthodes comme Word2vec [3] ou GloVe [1] utilisaient l'apprentissage d'un vecteur pour chaque mot. Aujourd'hui, les nouveaux modèles sont entraînés pour produire des représentations contextuelles. Chaque token sera représenté par un vecteur qui dépend du contexte à gauche et à droite. Ainsi, chaque représentation vectorielle d'un élément dépend de la phrase dans son entièreté. De 2015 à 2018, la tendance était d'utiliser des Réseaux de Neurones Récurrents (RNN). Ces réseaux sont adaptés pour des données d'entrée de taille variable et sont capables de conserver des informations en mémoire grâce aux connexions récurrentes. Pour cette raison, les RNNs sont particulièrement adaptés aux applications faisant intervenir le contexte, c'est à dire quand les données forment une suite et ne sont pas indépendantes les unes des autres, comme une suite de mots qui compose une phrase.

Aujourd'hui, des modèles comme GPT-2 [4], XLNet [5], BERT [6] et RoBERTa [7] utilisent la notion de Transformer au lieu des convolutions [2] ou de la récurrence. Ces modèles utilisent le mécanisme d' « attention » pour augmenter leur rapidité d'entraînement, notamment grâce à une meilleure parallélisation. Le concept d'attention a été mis en avant grâce au célèbre papier *Attention is All You Need* [8]. La figure 4.1 [3] ci-dessous présente le mécanisme d'attention pour le mot *it* de la phrase *The animal didn't cross the street because it was too tired*. L'attention établit des liens entre le mot qui est procédé et les mots jugés pertinents.

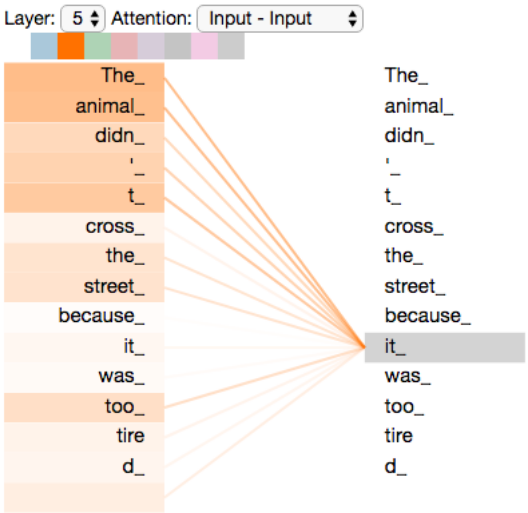


FIGURE 4.1: Encodage du mot *it* dans un Transformer. Le mécanisme d'attention se concentre principalement sur *The Animal*

Effectuer du transfert d'apprentissage à partir de ces modèles pré-entraînés a permis l'amélioration d'une grande variété de tâches de NLP. De plus, une grande partie des poids des modèles pré-entraînés est rendue publique, ce qui a permis de réduire considérablement le temps et les ressources pour l'analyse de données. Nous allons élaborer dans la prochaine section un modèle de classification des textes en anglais basé sur les Transformers.

1. En pratique, la plupart des chercheurs semblent effectuer un peu de pré-traitement et nettoyage sur les données.
2. Les convolutions sont très utilisées en vision par ordinateur.
3. `http://jalammar.github.io/illustrated-transformer/`

4.2 Modèle de classification des textes en anglais

4.2.1 Ensemble de données du HuffPost

Nous utilisons un jeu de données composé d'articles de HuffPost [1], un site web d'information aux États-Unis. Celui-ci rassemble plus de 200 000 articles récupérés de 2012 à 2018 par Rishabh Misra [2] [9]. Les données sont disponibles en téléchargement libre sur Kaggle [3]. Nous pouvons trouver pour chaque article son titre, sa description, son auteur, sa date de création et sa catégorie. Au total, il existe une vingtaine de catégories : *wellness*, *group voices*, *business & finance*, *parenting*, *world news*, *style & beauty*, *environment*, *food & drink*, *science & tech*, *arts culture*, *education*, *miscellaneous*, *crime*, *impact*, *women*, *home & living*, *sports*, *comedy*, *travel*, *entertainment*, *politics*. Certaines catégories initiales ont été rassemblées ou supprimées, comme *divorce*, *religion* et *weddings* selon les besoins de Makezu. Par exemple, déterminer la religion d'une personne enfreint les règles de Twitter. La figure 4.2 ci-dessous présente les catégories et leur distribution, nous voyons qu'elles ne sont pas uniformément représentées ce qui pourrait poser problème. Dans la prochaine sous-section, nous allons construire un modèle de classification permettant de catégoriser du texte selon les 20 classes retenues. L'entrée de notre modèle sera composé de la concaténation du titre et de la description de l'article avec sa catégorie correspondante. Nous faisons l'hypothèse que le titre et la description d'un article est similaire à la structure d'un tweet.

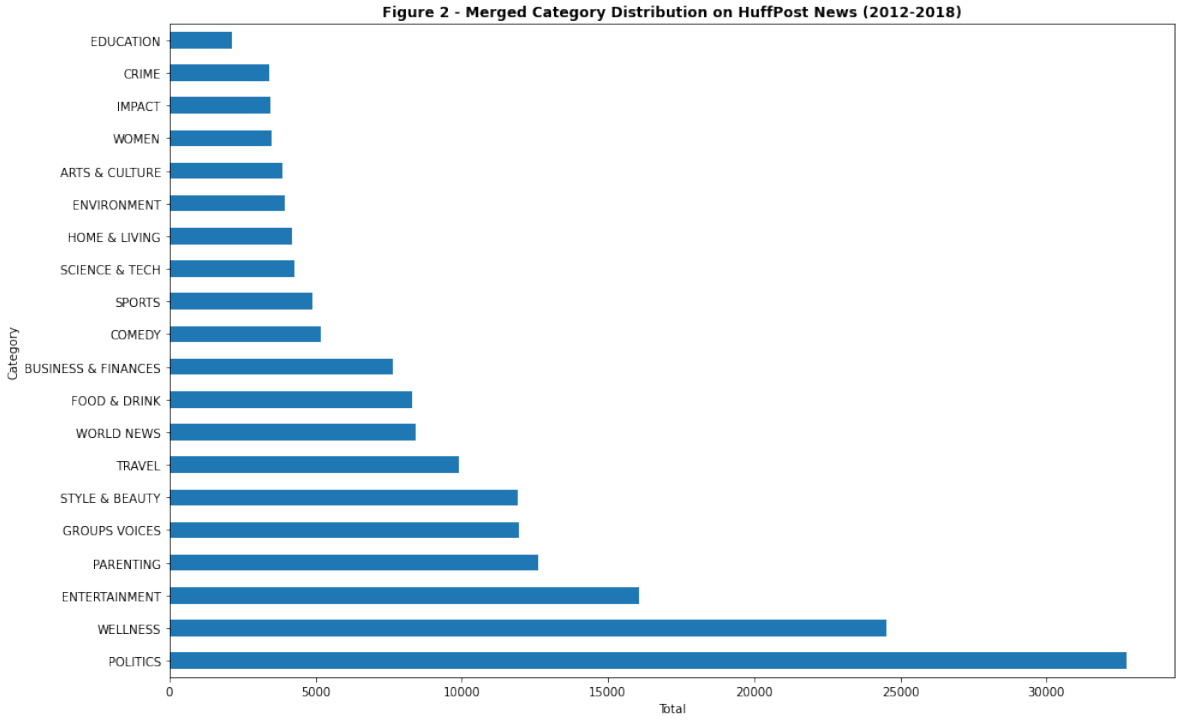


FIGURE 4.2: Distribution des catégories pour les articles du HuffPost récoltés

1. `https://www.huffpost.com`
2. https ://rishabhmisra.github.io/publications/
3. `https://www.kaggle.com/rmisra/news-category-dataset`

4.2.2 Présentation de BERT et RoBERTa

BERT, acronyme anglais de *Bidirectional Encoder Representations from Transformers*, est un modèle de langage développé par Google en 2018. Cette méthode a permis d'améliorer significativement les performances en traitement automatique des langues. BERT utilise les Transformers et permet d'apprendre, grâce à un mécanisme d'attention, les relations contextuelles entre les mots d'un texte. Contrairement aux modèles directionnels qui lit le texte séquentiellement, l'encodeur du Transformer lit la phrase dans son entièreté, il est donc considéré comme bidirectionnel. BERT utilise deux stratégies d'entraînement :

— le *Masked Language Model* (MLM). Avant d'envoyer les séquences de mots dans BERT, une partie des mots de chaque séquence est remplacée aléatoirement par le token [MASK]. Ainsi, le modèle doit apprendre la vraie valeur de [MASK] en se basant sur le contexte des autres mots qui constituent la séquence. Exemple : Pour la phrase *I [MASK] a student*, BERT doit prédire [MASK] = *am* ;
— la *Next Sentence Prediction* (NSP). Durant l'entraînement, 50% des entrées arrivent de pair avec la séquence qui les suit dans le document original, alors que 50% est une séquence aléatoirement choisie dans le corpus. Le but est de prédire si la seconde séquence est connectée à la première.

Lors de l'entraînement d'un modèle de BERT, MLM et NSP sont entraînés ensemble, le but est de minimiser la fonction de perte de ces deux stratégies.

A Robustly Optimized BERT Pretraining Approach (RoBERTa), développé par Facebook, est construit à partir de l'architecture de BERT mais le pré-entraînement a été optimisé. La valeur des hyperparamètres ont été modifiée comme le *learning rate* et le *batch size*[1] qui sont désormais plus élevés. Le modèle est également pré-entraîné sur un plus large corpus de texte (160 GB). Pour finir, une seule stratégie d'entraînement a été gardée, qui est le MLM. La prochaine section expose comment un modèle RoBERTa est affiné pour la catégorisation des tweets.

4.2.3 Affinement du modèle BERT

Pour affiner un modèle BERT, nous utiliserons les librairies suivantes :

— **transformers**[2] de Huggingface. Cette librairie construite par la société franco-américaine Huggingface permet d'entraîner des modèles atteignant l'état de l'art pour du traitement du langage naturel. Elle fournit de nombreuses architectures comme BERT, GPT-2, RoBERTa et DistilBERT. De nombreux modèles pré-entraînés et des ensembles de données sont disponibles gratuitement. Les modèles entraînés avec cette librairie peuvent fonctionner sur TensorFlow et PyTorch. PyTorch est une bibliothèque Python *open source* d'apprentissage machine qui s'appuie sur Torch développée par Facebook. PyTorch permet d'effectuer des calculs tensoriels, notamment pour de l'apprentissage profond. TensorFlow, développée par Google, est similaire au fonctionnement de PyTorch. Nous utiliserons Transformers pour initialiser et entraîner notre modèle ;
— **scikit-learn**[3]. Cette bibliothèque libre Python est destinée à l'apprentissage automatique. Elle est développée par de nombreux contributeurs et est très utilisée dans le monde académique. Nous l'utiliserons pour uniformiser l'importance des classes de notre modèle, diviser l'ensemble de données entre les données d'entraînement (*training data*) et les données de test (*test data*) ainsi que pour avoir certaines métriques ;
— **pandas**[4]. Une librairie Python *open source* permettant la manipulation des données. Nous l'utiliserons sur quasiment tous nos projets ;

1. Nous verrons plus en détail ce que signifie ces deux termes lors de l'entraînement de notre modèle.
2. `https://huggingface.co`
3. `https://scikit-learn.org/`
4. `https://pandas.pydata.org`

— **numPy** [1]. La bibliothèque permet d'effectuer du calcul numérique avec Python, elle facilite notamment le calcul matriciel. Comme pour Pandas, nous l'utiliserons sur quasiment tous nos projets.

L'entraînement du modèle s'effectue dans un environnement Google Colab avec calcul sur TPU. Le TPU, pour unité de traitement de tenseur, est un circuit intégré développé par Google spécifiquement pour accélérer les systèmes d'intelligence artificielle par réseaux de neurones. En effet, ces réseaux font appel à énormément de calcul tensoriel. Comme nous allons le voir, les TPU améliorent significativement la rapidité d'entraînement des modèles.

Des points de sauvegarde sont effectués à chaque époque (*epoch*) sur mon environnement Google Drive. La valeur d'époque détermine le nombre de fois où l'ensemble du jeu de données passe et met à jour le modèle. Étant donné que les classes du jeu de données ne sont pas uniformément représentées, comme nous l'avons vu sur la figure 4.2, nous générons avec **scikit-learn** des poids qui donneront plus d'importance aux classes sous-représentées, inversement, les classes sur-représentées auront un poids moins important durant l'entraînement.

Le *dropout* consiste à aléatoirement désactivé une portion des neurones durant l'entraînement. Cela permet d'agrandir l'espace de recherche et de ne pas tomber dans des optima locaux, ce qui empêche théoriquement le surentraînement (*overfitting*). Le papier *Multi-Sample Dropout for Accelerated Training and Better Generalization* [10] présente une technique de *dropout* avancé qui permet d'accélérer l'entraînement tout en améliorant la généralisation du modèle. La méthode consiste à faire passer la sortie du Transformer dans plusieurs couches de *dropout* en parallèle. La perte finale est la moyenne des pertes de chaque couche. Ainsi, cela permet de stabiliser l'entraînement. Nous avons implémenté l'architecture illustrée sur la figure 4.3 pour l'entraînement de notre modèle.

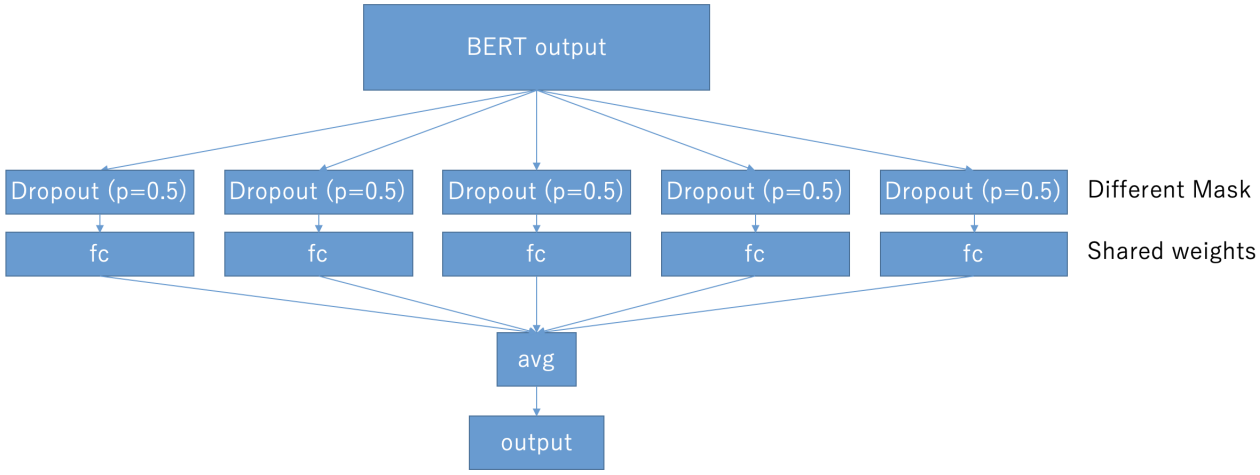


FIGURE 4.3: Cinq couches de *dropout* avec 50% de chance qu'un neurone ne soit pas activé. Chaque couche est suivie d'une couche entièrement connectée (*fully connected layer*)

L'idée du *multi-sample dropout* provient de la solution de l'équipe ayant obtenu la huitième place à un concours Kaggle [2]. Le but était d'élaborer un modèle de classification permettant de détecter la toxicité dans les conversations.

Vous trouverez ci-dessus un tableau récapitulatif des différents hyperparamètres utilisés pour entraîner le modèle. Le *batch_size* correspond à la taille du lot d'échantillon à envoyer dans le

1. `https://numpy.org`
2. `https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification/discussion/100961`

Hyperparamètre	Valeur
train_test_split	80/20%
train_batch_size	32*nb_replicas_in_sync
eval_batch_size	32*nb_replicas_in_sync
epoch	10
learning_rate	$3 * 10^{-5}$
dropout_rate	50%

TABLE 4.1: Hyperparamètres du modèle RoBERTa

modèle à chaque itération. Le *learning rate* est la vitesse d'apprentissage du modèle, il contrôle le pas de mise à jour des poids du modèle en réponse à une erreur estimée. Si le paramètre est trop petit, l'entraînement durera trop longtemps, en revanche, s'il est trop grand, le modèle risque de se bloquer dans un optimum local. Nous choisissons une valeur recommandée pour entraîner des modèles BERT. Les données d'entraînement représentent 80% de l'ensemble des données. Nous utilisons les 20% restants pour évaluer le modèle et ainsi avoir une idée de sa robustesse. Un modèle est robuste lorsqu'il arrive à prédire correctement des données qu'il n'a pas encore observé.

Les multiples coeurs du TPU permettent de paralléliser l'entraînement. En effet, nous pouvons envoyer des lots de données sur chaque coeur. La taille du lot a été fixée à 32 sur chaque coeur et le TPU mis à disposition par Google possède 8 coeurs. Nous entraînons donc notre modèle avec un lot de $32 * 8 = 256$ d'articles par itération. Il faut seulement 170 secondes pour entraîner le modèle sur la totalité de l'ensemble d'entraînement, ce qui est exceptionnel étant donné la taille des données. À partir de la sixième époque, le modèle devient moins robuste et est moins performant sur l'ensemble d'évaluation. La cinquième époque fournit la meilleure précision qui est de 0.8094. Le modèle prédit la catégorie de 80% des articles correctement [1], ce qui est un très bon score pour un modèle ayant une vingtaine de classes différentes. Pour finir, un modèle est entraîné sur la totalité des données avec cinq époques. Le temps d'entraînement n'a été que de 973 secondes. Dans la section 4.5, des approches sont présentées pour faciliter la mise en production du modèle pour Makezu, comme la distillation ou la quantification. Dans la prochaine section, nous construisons « de zéro » un jeu de données qui sera utilisé pour entraîner un modèle de classification en français.

1. Bien entendu, la précision est calculée sur l'ensemble de données d'évaluation.

4.3 Élaboration d'un jeu de données composé d'articles francophones de journaux en ligne

4.3.1 Programme de récupération des articles

Comme nous l'avons vu dans la précédente section, il est important d'avoir une grande quantité d'articles avec leur catégorie correspondante pour pouvoir construire un modèle de classification. Bien qu'il existe un tel jeu de données en anglais, il n'existe pas à l'heure actuelle de jeux de données similaires en français. Une solution temporaire est de traduire en anglais le texte à prédire provenant d'autres langues et d'utiliser le modèle de classification en anglais. Cela demande peu de ressources mais ce n'est en pratique pas une solution envisageable pour Makezu. En effet, il faut pouvoir prédire des centaines de tweets, le temps d'inférence du modèle n'est pas un problème, mais le temps de traduction représente un coût conséquent. Ainsi, il existe deux solutions :

— traduire le jeu de données anglais vers le français en passant par exemple par l'API de Google Traduction [1]. Cependant, la traduction n'est jamais parfaite et cela impactera la précision du modèle de classification. De plus, il y aura un manque de « contexte français », des entités comme les lieux ou les noms propres français peuvent être très utiles pour prédire la catégorie d'un texte. La traduction ne permet pas d'avoir ces éléments ;

— construire notre propre jeu de données français. Le temps de travail est plus conséquent mais le modèle aura une bien meilleure précision.

J'ai choisi la seconde option puisqu'elle semble alignée avec les demandes de Makezu. Il est nécessaire d'avoir une bonne précision des modèles ainsi que des procédés qui peuvent facilement se mettre à l'échelle. Le but est donc de récupérer des articles francophones [2] en ligne, en particulier les titres et descriptions pour élaborer un jeu de données français sur le même modèle que le jeu de données anglais présenté dans la section précédente [3]. Cela permettra de construire des modèles français de classification de texte.

J'utilise la librairie Python **newspaper** [4] qui permet de récupérer la liste contenant les URLs des articles pour un journal en ligne donné. Ensuite, j'utilise la librairie Python **news-please** [5] pour l'extraction des informations comme le titre ou la description d'un article à partir d'une URL donnée. Le programme fonctionne pour une très grande majorité des journaux francophones. Par exemple, cela fonctionne pour des journaux « classiques » comme Le Monde, Le Figaro, Liberation ou Le Parisien mais aussi pour des journaux un peu plus spéciaux comme La Recherche et Sciences et Avenir pour du contenu scientifique ou Le Gorafi et Nordpress pour du contenu humoristique. L'intérêt est d'avoir un large spectre de journaux similaires aux divers contenus textuels que l'on peut trouver sur internet, et notamment sur Twitter. Au total, je récupère les articles de 69 différents journaux en ligne.

Nous voyons sur la figure 4.4 que le programme opère en différentes étapes. Tout d'abord, il se connecte sur un journal et charge le fichier CSV qui contient les articles précédemment récupérés. Ensuite, il explore les nouveaux articles qui n'ont pas encore été récupérer. Le programme passe à un autre journal si :

— tous les nouveaux articles ont été récupérés ;

— le journal actuel ne renvoie pas une réponse correcte (une valeur non égale à 200). Cela veut dire que le programme ne peut plus récupérer d'articles. Cela est principalement due aux

1. `https://cloud.google.com/translate/?hl=fr`
2. Les articles proviennent de sources françaises et belges francophones.
3. `https://www.kaggle.com/rmisra/news-category-dataset`
4. `https://github.com/codelucas/newspaper`
5. `https://github.com/fhamborg/news-please`

limitations internes de requêtes imposé par le site. Par exemple, si le programme envoie trop de requêtes, son adresse IP peut être bloquée temporairement.

FIGURE 4.4: Sortie tronquée du programme qui récupère les articles

Le programme s'arrête lorsqu'il a parcouru tous les journaux. Reprenons l'exemple décrit sur la figure ci-dessus. Le programme se connecte sur `slate.fr`, charge les 3537 articles déjà récupérés, découvre qu'il y a 138 nouveaux articles et les récupère tous. Il passe ensuite à `latribune.fr` et procède de la même façon. Cependant, il se fait bloquer à partir de 70 articles récupérés. Pour chaque article, la date de création, les auteurs, le titre, une description courte, le contenu principal et l'URL sont enregistrés. Vous trouverez en annexe 4.9 une affichage tronqué du fichier CSV contenant la totalité des articles récupérés. Le contenu principal est souvent tronqué durant la récupération puisque de plus en plus de journaux affichent la totalité de leurs articles à condition que l'utilisateur paie un abonnement. Chaque jour, entre 3000 et 6000 articles sont récupérés en une centaine de minutes. Le programme est lancé dans un environnement Google Colab depuis le mois d'octobre. Les fichiers CSV de chaque journal sont enregistrés sur un environnement de stockage Google Drive.

4.3.2 Annotation des articles

Pour élaborer un modèle de classification, il faut des données annotées. Maintenant que nous avons récupérer les articles, il faut les annoter pour ainsi lier le contenu textuel à la catégorie correspondante. Une première méthode est d'extraire les méta-données d'un article et d'espérer trouver des informations sur la catégorie. Cette méthode n'est pas fiable car trop peu de journaux en ligne renseignent ce genre d'information. Néanmoins, il existe une autre manière d'annoter automatiquement et très rapidement les données. La structure d'une URL possède des informations intéressantes. Les URLs des articles sont utilisées pour annoter automatiquement les catégories correspondantes. Voici un exemple pour l'article : *Qu'est-ce qu'être français ? : Emmanuel Macron engage le débat sur l'identité*. L'URL correspondante est https ://www.lemonde.fr/**politique**/article/2020/12/22/qu-est-ce-qu-etre-francais-emmanuel-macron-engage-le-debat-sur-l-identite.html qui contient le mot « politique ». Ainsi, l'article aura comme classe « politique ». Avec la librairie Python **re** [1] permettant de faire des manipulations regex, nous pouvons savoir quelles sont les catégories les plus représentées dans les URLs. Une sortie tronquée est illustrée dans la figure 4.5. Par exemple, 6691 URLs contiennent le mot « santé ». Il est intéressant de voir que le terme « coronavirus » apparaît 2352 fois. Un sous-ensemble d'articles concernant le coronavirus pourrait être intéressant pour alimenter la recherche, notamment sur la réception médiatique d'un tel événement.

1. Ce module fournit des opérations sur les expressions rationnelles.

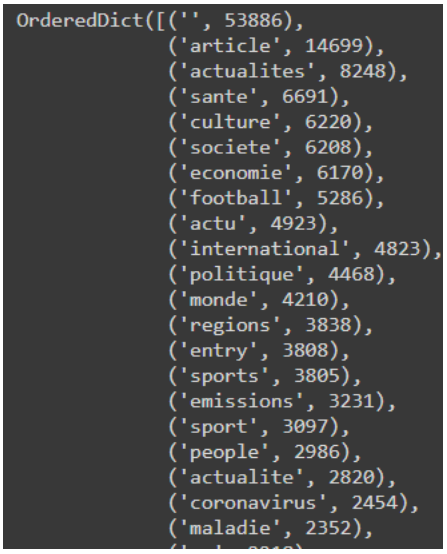


FIGURE 4.5: Sortie tronquée des catégories les plus représentées

Nous retenons dix classes qui pourraient englober la totalité des articles. Par exemple, la classe « économie » contient les sous-catégories : argent, finance, économie, bourse, entreprise, emploi, monnaie, startup, innovation, consommation, business, conso, eco, banque, assurance, marché, financier, impôt et carrière. Le même procédé est effectué pour les neuf autres classes. Un récapitulatif du nombre d'occurrence des classes dans le jeu de données est illustré dans la tableau ci-dessous.

Classe	Nombre d'occurrence
Économie	16163
Science	6365
Politique	7756
Culture	17688
Sport	12088
Santé	8260
International	11519
Environnement	4503
Style	10446
Société	13171

TABLE 4.2: Le nombre d'occurrence pour chaque classe

4.3.3 Pistes d'amélioration

Apprentissage semi-supervisé

Au total, sur 166880 articles récupérés, notre méthode retourne 107959 annotations. Cependant, un article peut appartenir à plus d'une classe à la fois selon la structure de son URL, ce qui concerne un dixième du jeu de données. Ainsi, avec la méthode d'annotation automatique à travers les URLs, nous pouvons annoter la moitié de la totalité des articles. L'autre moitié pourra être annoté avec d'autres méthodes. L'apprentissage semi-supervisé est une classe de techniques d'apprentissage automatique qui utilise un ensemble de données annotées et non annotées pour

améliorer significativement la qualité de l'apprentissage. Avec notre ensemble de données, nous pouvons effectuer une des méthodes d'apprentissage semi-supervisé qui est l'apprentissage actif. Une stratégie est par exemple de choisir les données dont le modèle a le plus de doute sur l'annotation en calculant par exemple l'entropie [1]. Il suffit alors d'annoter « à la main » ces données pour améliorer les performances du modèle. Pour notre cas, cette méthode ne semble pas être efficace puisqu'elle est surtout utilisée lorsque l'ensemble de données est très petit, ce qui n'est pas notre cas. Une autre méthode d'apprentissage semi-supervisé semble correcte par rapport à notre jeu de données, le *pseudo-labeling*. Cela consiste à entraîner un modèle de classification avec les données annotées, puis utiliser ce modèle pour prédire les classes des données non-annotées. Pour ne pas perdre en performance, il est recommandé d'annoter une donnée seulement si le modèle est très confiant sur sa classe, selon un seuil que l'on définit au préalable. Pour finir, il suffit d'entraîner le modèle final sur les données annotées et pseudo-annotées.

Autres utilisations

Le jeu de données peut être utiliser pour d'autres buts que l'élaboration de modèles de classification. Nous pouvons faire de la recommandation d'articles avec par exemple l'*Universal Sentence Encoder* [11], un modèle multi-langue de Google permettant l'encodage de phrase en vecteurs, très utile pour travailler sur la similarité sémantique. Il pourrait également être utiliser dans un but malveillant, pour diffuser de la désinformation en générant de faux articles de journaux. Il suffit d'entraîner un modèle de langage permettant la génération de texte comme GPT-2 [4], un modèle construit par la société OpenAI entraîné sur 8 millions de pages web et contenant 1.5 milliards de paramètres [2].

Pour finir, il serait intéressant d'alimenter le jeu de données avec d'autres types de donnée comme par exemple du contenu textuel provenant de Reddit et de Quora. En effet, ces plateformes possèdent des contraintes d'écriture différentes par rapport à Twitter. Par exemple, la limitation du nombre de caractères par message n'est pas la même. Les différentes plateformes sociales sont des supports techniques qui modifient nos manières d'écrire et de penser. Si nous voulons comprendre et modéliser le langage, il est indéniable qu'il faudra croiser les données en utilisant différentes sources. Les réflexions philosophiques autour de l'écriture numérique sont au coeur des discussions du laboratoire Costech de l'université de technologie de Compiègne. Par exemple, il serait intéressant d'étudier les écrits de Bernard Stiegler et la théorie du support de Bruno Bachimont pour mieux comprendre les modèles de langage. Dans la prochaine section, un modèle de classification des textes en français est construit grâce à notre nouveau jeu de données.

1. L'entropie est une mesure d'incertitude à partir de l'ensemble des probabilités d'appartenance aux classes.
2. https ://openai.com/blog/better-language-models/

4.4 Modèle de classification des textes en français

Maintenant que nous avons un ensemble de données en français, nous pouvons entraîner un modèle de classification qui permettra de prédire la catégorie d'un texte selon une dizaine de classe : culture, économie, environnement, monde, politique, santé, science, société, sport et style. Pour rappel, parmi les informations récoltées sur les articles, nous nous intéressons seulement aux titres et aux descriptions. L'entrée de notre modèle sera composée de la concaténation du titre et de la description de l'article avec sa classe correspondante. Ce modèle aura pour but d'être utiliser pour classifier des tweets et ainsi analyser les tendances ainsi que les profils sur Twitter. Nous faisons donc l'hypothèse que le titre et la description d'un article est similaire à la structure d'un tweet.

Le modèle de BERT étant pré-entraîné sur du texte anglais, il n'est pas pas très rigoureux d'affiner ce modèle sur nos données françaises. À la place, nous allons utiliser deux modèles, FlauBERT [12] et CamemBERT [13], construits spécifiquement pour des tâches françaises et atteignant l'état de l'art des modèles de langage français.

4.4.1 Présentation de FlauBERT

Une grande majorité des modèles sont pré-entraînés sur du texte anglais. Il existe des versions multi-langues qui prennent en compte plusieurs dizaines de langues en un seul modèle, comme mBERT, XLM [14] et XLM-R [15]. Néanmoins, FlauBERT est plus performant sur des tâches spécifiques de NLP en français que les modèles multi-langues.

FlauBERT est un modèle de BERT entraîné sur un très large et hétérogène corpus de texte en français. Au total, 24 sous-corpus ont été rassemblés provenant de différentes sources disponibles gratuitement [1] et couvrant divers sujets et styles d'écriture. Cela correspond à plus de 270 GB de texte, soit 71 GB après nettoyage et application de filtres. Les modèles de différentes tailles (*small, base* [2]*, large* [3]) sont entraînés en utilisant le nouveau super-ordinateur du CNRS, Jean Zay [4]. Les poids des modèles sont disponibles en ligne [5].

4.4.2 Affinement du modèle FlauBERT

Nous utilisons la librairie python **simpletransformers** [6] qui est basée sur la librairie **transformers** de HuggingFace. Simpletransformers permet d'initialiser, entraîner et évaluer rapidement des modèles Transformers grâce à un haut niveau d'encapsulation. Ces librairies sont très utiles en entreprise puisque cela permet de lancer rapidement des modèles en production, néanmoins, l'opacité technique augmente et nous pensons qu'un ingénieur doit maîtriser la technicité de ses outils dans la mesure du possible. Étant donné que nous ne voulons pas modifier l'architecture du modèle mais seulement se concentrer sur la bonne valeur des hyperparamètres, cette librairie est intéressante.

L'entraînement du modèle s'effectue dans un environnement Google Colab avec calcul sur GPU, une Tesla P100. Des points de sauvegarde sont effectuer à chaque époque sur mon environnement Google Drive. Étant donné que les classes du jeu de données ne sont pas uniformément représentées, nous générons avec la librairie python **scikit-learn** des poids qui donneront plus d'importance aux

1. Les sources sont des livres, Wikipedia, Common Crawl etc.
2. Le modèle *base* contient 138 millions de paramètres.
3. Le modèle *large* contient 373 millions de paramètres.
4. `http://www.idris.fr/eng/jean-zay/`
5. `https://github.com/getalp/Flaubert`
6. `https://github.com/ThilinaRajapakse/simpletransformers`

classes sous-représentées, inversement, les classes sur-représentées auront un poids moins important durant l'entraînement.

Classe	Poids
Économie	0.67
Science	1.71
Politique	1.58
Culture	0.57
Sport	0.75
Santé	1.74
International	0.94
Environnement	2.21
Style	1.06
Société	0.82

TABLE 4.3: Poids assigné pour chaque classe

Nous utilisons la métrique *Label ranking average precision* (LRAP) [1] comme indicateur de performance de notre modèle. Cette métrique est utilisé lorsque nous faisons face à un problème de rang des différentes classes, ce qui est notre cas puisqu'une donnée peut appartenir à plusieurs classes, comme nous l'avons vu dans la sous-section 4.3.2 sur l'annotation automatique des articles. Le but est de donner un meilleur rang aux classes associées à chaque échantillon. Le score obtenu est toujours supérieur à 0 et sa plus haute valeur est 1.

Vous trouverez ci-dessous un tableau récapitulatif des différents hyperparamètres utilisés pour entraîner le modèle. Le *batch_size* correspond à la taille du lot d'échantillon à envoyer dans le modèle à chaque itération. Nous avons choisi la valeur 32 pour optimiser l'allocation des ressources du GPU. Le *learning rate* est la vitesse d'apprentissage du modèle, il contrôle le pas de mise à jour des poids du modèle en réponse à une erreur estimée. Si le paramètre est trop petit, l'entraînement durera trop longtemps, en revanche, s'il est trop grand, le modèle risque de se bloquer dans un optimum local. Nous choisissons une valeur recommandée pour entraîner des modèles BERT. La valeur d'époque détermine le nombre de fois où l'ensemble du jeu de données passe et met à jour le modèle. Les données d'entraînement représentent 80% de l'ensemble des données. Nous utilisons les 20% restant pour évaluer le modèle et ainsi avoir une idée de sa robustesse. Un modèle est robuste lorsqu'il arrive à prédire correctement des données qu'il n'a pas encore observé.

Hyperparamètre	Valeur
train_test_split	80/20%
train_batch_size	32
eval_batch_size	32
epoch	5
learning_rate	$3 * 10^{-5}$

TABLE 4.4: Hyperparamètres du modèle FlauBERT

Le modèle FlauBERT entraîné sur nos données provenant d'articles français est correctement affiné au bout de deux époques. En effet, après la deuxième époque, la valeur retournée par la fonction de perte lors de l'évaluation du modèle augmente, ce qui montre que le modèle commence

1. Pour plus d'informations sur la façon dont on calcule l'indice, voir la documentation de **sklearn** : `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.label_ranking_average_precision_score.html`

à sur-apprendre. Il commence à connaître par coeur les données d'apprentissage et perd en robustesse [1]. L'indice LRAP est de 0.8625, ce qui est un très bon score. Ainsi, pour Makezu, un nouveau modèle est entraîné sur deux époques sur la totalité du jeu de données. Dans la prochaine sous-section, un autre modèle français, CamemBERT, est entraîné sur notre ensemble de données.

4.4.3 Présentation du modèle CamemBERT

CamemBERT, développée par l'INRIA [2], est un modèle basé sur l'architecture RoBERTa [7] entraîné sur 282 GB de texte français, soit 138 GB après nettoyage et application des filtres. Les données françaises proviennent du corpus multi-langue OSCAR [3], un corpus filtré de Common Crawl [4], très utilisé pour effectuer de l'apprentissage non-supervisé de modèles de langage. Un fait surprenant mentionné dans le papier CamemBERT, le modèle entraîné sur un échantillon de 4 GB de texte d'OSCAR est similaire au niveau des performances que le modèle standard entraîné sur la totalité (138 GB). Le modèle final CamemBERT *base* contient 110 millions de paramètres.

4.4.4 Affinement d'un modèle CamemBERT

Nous utilisons, comme pour FlauBERT, la librairie **simpletransformers** pour élaborer et entraîner le modèle CamemBERT. L'entraînement du modèle s'effectue dans un environnement Google Colab avec calcul sur GPU, une Tesla P100. Des points de sauvegarde sont effectuer à chaque époque sur mon environnement Google Drive. Nous utilisons lors de l'entraînement les mêmes poids de classe mentionnés dans le tableau 4.3 ainsi que les mêmes hyperparamètres mentionnées dans le tableau 4.4.

Le modèle camemBERT entraîné sur nos données provenant d'articles français est correctement affiné au bout de quatre époques. En effet, après le quatrième époque, la valeur retournée par la fonction de perte lors de l'évaluation du modèle augmente, ce qui montre que le modèle commence à sur-apprendre. L'indice LRAP est de 0.8629, ce qui est un très bon score. Celui-ci est sensiblement supérieur à l'indice LRAP de Flaubert qui est de 0.8625, nous pouvons néanmoins considérer les deux modèles comme ayant des performances équivalentes. Pour Makezu, un nouveau modèle est entraîné sur quatre époques sur la totalité du jeu de données.

Si nous voulons des performances optimales pour la prédiction des tweets, nous pouvons utilisons un ensemble des deux modèles FlauBERT et CamemBERT. On peut par exemple faire passer un tweet dans les deux modèles et établir une moyenne des prédictions des deux modèles. Néanmoins, le coût computationnel est trop important, la mise en production des modèles est une étape cruciale pour une entreprise, c'est ce que nous allons voir dans la prochaine section.

4.5 Mise en production des modèles pour Makezu

4.5.1 Quantification des modèles

L'utilisation des modèles pour Makezu pose un problème technique qui est celui du temps d'inférence. En effet, ces modèles demandent énormément de ressources, et même avec un GPU, il est coûteux de procéder des milliers de tweets. Il existe une méthode pour réduire la taille du modèle et le temps d'inférence en sacrifiant un peu de précision, c'est la quantification. La quantification consiste à approximer les poids d'un modèle. Il existe deux sortes de quantification. La première

1. Le phénomène connu d'*overfitting*.
2. Institut national de recherche en sciences et technologies du numérique.
3. `https://oscar-corpus.com`
4. Common Crawl est un répertoire ouvert de données récupérées sur des pages web : `https://commoncrawl.org`

est la quantification durant l'entraînement, nous pouvons entraîner par exemple deux réseaux de neurones, un pour la précision flottante et une pour la précision binaire des poids. La figure 4.6 [1] ci-dessous illustre le procédé : durant la « passe en avant » (*forward pass*), la précision binaire est mise à jour en utilisant la précision flottante. Durant la « passe en arrière » (*backward pass*), la précision flottante est mise à jour en utilisant la précision binaire. L'entraînement résout un problème d'optimisation alternée. Le modèle binaire est ensuite utilisé pour l'inférence.

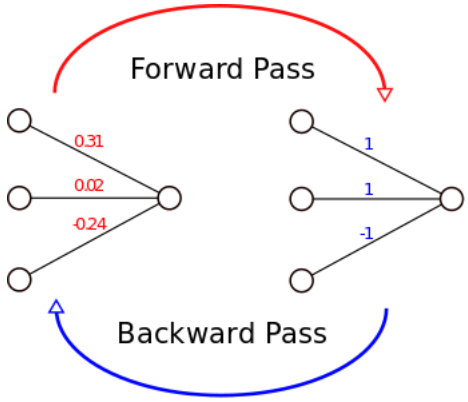


FIGURE 4.6: Quantification des poids durant l'entraînement

Il existe également la quantification post-entraînement. Celle-ci est moins efficace que la quantification durant l'entraînement, néanmoins elle est beaucoup plus simple à mettre en place. Par exemple, la quantification d'entiers est une stratégie d'optimisation qui convertit les nombres à virgule flottante de 32 bits (tels que les poids et les sorties d'activation) en nombres à virgule fixe de 8 bits les plus proches. Cela se traduit par un modèle quatre fois plus petit ainsi qu'une vitesse d'inférence trois fois plus rapide. Il existe également la conversion des poids en valeurs à virgule flottante 16 bits qui permet de diviser par deux la taille du modèle. Bien entendu, il faut s'assurer que la dégradation est acceptable et ne baisse pas la précision du modèle de manière significative. Nous allons voir une autre approche qui permet également de réduire la taille et le temps d'inférence, la distillation.

4.5.2 Distillation des modèles

La distillation des connaissances consiste à compresser un large modèle, comme BERT, en un plus petit modèle. Ce n'est pas seulement un changement de poids, mais aussi un changement de l'architecture global. L'idée est d'entraîner un plus petit modèle, l'élève, en utilisant le modèle large déjà entraîné, le professeur. Le modèle élève est entraîné pour avoir le même comportement que le modèle professeur en essayant de reproduire les sorties à chaque niveau de l'architecture, et pas seulement la sortie finale. Ainsi, pour améliorer le temps d'inférence et la taille du modèle anglais construit sur l'architecture de RoBERTa, nous pouvons entraîner un modèle avec distillation comme distilRoBERTa [16], une version distillée du modèle RoBERTa. La version distillée atteint 95% des performances sur GLUE [2] tout en étant deux fois plus rapide et 35% plus petit [3]. Il n'existe malheureusement pas de version distillée de CamemBERT et FlauBERT actuellement. En conclusion, si dans l'avenir Makezu doit procéder énormément de tweets tout en ayant des ressources

1. `https://medium.com/@joel_34050/quantization-in-deep-learning-478417eab72b`
2. GLUE (*General Language Understanding Evaluation*) est un benchmark permettant d'évaluer la compréhension du langage.
3. `https://code.ihub.org.cn/projects/763/repository/revisions/master/entry/examples/distillation/README.md`

computationnelles réduites, il est recommandé de combiner la distillation et la quantification des modèles.

En conclusion, deux modèles atteignant l'état de l'art ont été construits pour pouvoir catégoriser le contenu textuel sur Twitter, en anglais et en français. Un jeu de données composé d'articles francophones a été élaboré pour pouvoir entraîner le modèle français. Ces modèles permettront d'améliorer l'analyse des tendances et le profilage des utilisateurs pour Makezu.

Conclusion

Nous avons pu améliorer l'application Makezu de différentes manières. Les outils d'extraction de données permettront d'élargir le champ d'action de Makezu à d'autres plateformes sociales comme Quora. Ces outils peuvent être utilisés pour automatiser le contenu des clients, comme nous l'avons montré en reliant Quora et Twitter. Nous avons également améliorer la capacité d'analyse de données de l'application en construisant des modèles qui atteignent l'état de l'art. Les tendances Twitter ainsi que les utilisateurs peuvent être analysés plus finement à l'aide de la reconnaissance des entités nommées et surtout de la classification des tweets en différentes catégories. La version française du dernier modèle ne pourrait fonctionner sans l'élaboration d'un jeu de données composé de dizaines de milliers d'articles francophones. L'enjeu futur est d'intégrer correctement ces nouveaux outils à l'application et de s'assurer qu'ils puissent correctement se mettre à l'échelle. Il me semble également utile d'entretenir une discussion sur l'éthique des modèles de traitement de données.

En effet, il est important de considérer le caractère normatif des objets techniques [1]. La méconnaissance de la technicité d'un outil mène à la méconnaissance des valeurs et des potentialités qui résident dans cet outil. Un outil n'est pas neutre, il ne se définit pas seulement par son utilisation. Il faut considérer la non-neutralité de la technique. Il est dangereux qu'une petite communauté de développeurs ait la main sur la construction de modèles ayant des impacts gigantesques dans nos sociétés. Par exemple, les requêtes sur le moteur de recherche Google sont désormais analysées par des modèles BERT, ce qui impactera les résultats des recherches dans plus de 70 langues [2]. Les larges modèles de langage amplifient considérablement la vision de ses constructeurs. Aujourd'hui, des modèles comme BERT ou GPT-2 posent de sérieux problèmes éthiques. Pour entraîner de tels modèles, les chercheurs collectent massivement des données provenant d'internet. Il y a un risque que du langage offensif, comme du langage sexiste ou raciste, soit incorporé dans les données d'entraînement. De plus, un modèle d'intelligence artificielle aura du mal à capturer les langues et les normes d'un pays et de personnes qui ont moins accès à internet. Les modèles de langage construisent un langage homogène, qui reflète les pratiques des communautés riches au dépend des communautés opprimées. De plus, étant donné la largeur des données d'entraînement, il est difficile de les auditer et de vérifier les biais embarqués. Nous recommandons la lecture du papier *Social Biases in NLP Models as Barriers for Persons with Disabilities* [17] qui met en lumière de multiples biais sociaux dans les modèles de langage récents, en étudiant notamment la prédiction de la toxicité et du sentiment d'un texte en rapport avec le handicap. Ainsi, mettre des technologies avancées d'intelligence artificielle à la main de marketeurs peut être un danger si ces derniers ne connaissent pas les potentialités des outils qu'ils utilisent.

Hélène et Hugo m'ont permis de m'épanouir au sein des projets grâce à la part importante qu'ils attribuent à la créativité et à l'avis de l'autre. La principale contrainte a été technique. Il était dommage de ne pas avoir eu à disposition des cartes graphiques (GPU) pour entraîner nos

1. Pour plus d'information : `http://www.costech.utc.fr/CahiersCOSTECH/spip.php?article28`
2. L'annonce de cette mise à jour a été faite sur le compte Twitter de Google : `https://twitter.com/searchliaison/status/1204152378292867074`

modèles mais aussi pour les lancer en production. Il était ainsi difficile de construire des modèles atteignant l'état de l'art et de rester compétitif.

Bien que j'étais déjà familier avec les méthodes d'apprentissage statistique, avant d'entrer chez Makezu je ne connaissais le traitement naturel du langage qu'en surface. J'ai pu approfondir ce domaine interdisciplinaire, à la croisé entre l'informatique, la linguistique et les statistiques. Pour la première fois, je me suis intéressé aux études récentes et j'ai lu mes premiers papiers scientifiques, ce qui m'a donné goût à la recherche. Je trouve fascinant la créativité des chercheurs pour explorer de nouvelles architectures. Ce qui m'intéresse principalement est la mise en communication du traitement du langage naturel avec le domaine de la vision par ordinateur, des concepts nouveaux peuvent émerger avec la mise en relation de domaines différents. L'aventure continue au prochain semestre, puisque j'effectuerai l'analyse exploratoire des réactions sur Twitter aux résultats d'affectation de Parcoursup [1] sous la direction d'Anne Bellon et Michael Vicente, deux professeurs à l'UTC. Je compte également entretenir mon site `gustavecortal.com`, qui est une plateforme où j'étudie principalement la relation entre l'intelligence artificielle et l'art.

1. Une plateforme web destinée à recueillir et gérer les vœux d'affectation des futurs étudiants de l'enseignement supérieur français.

Annexe

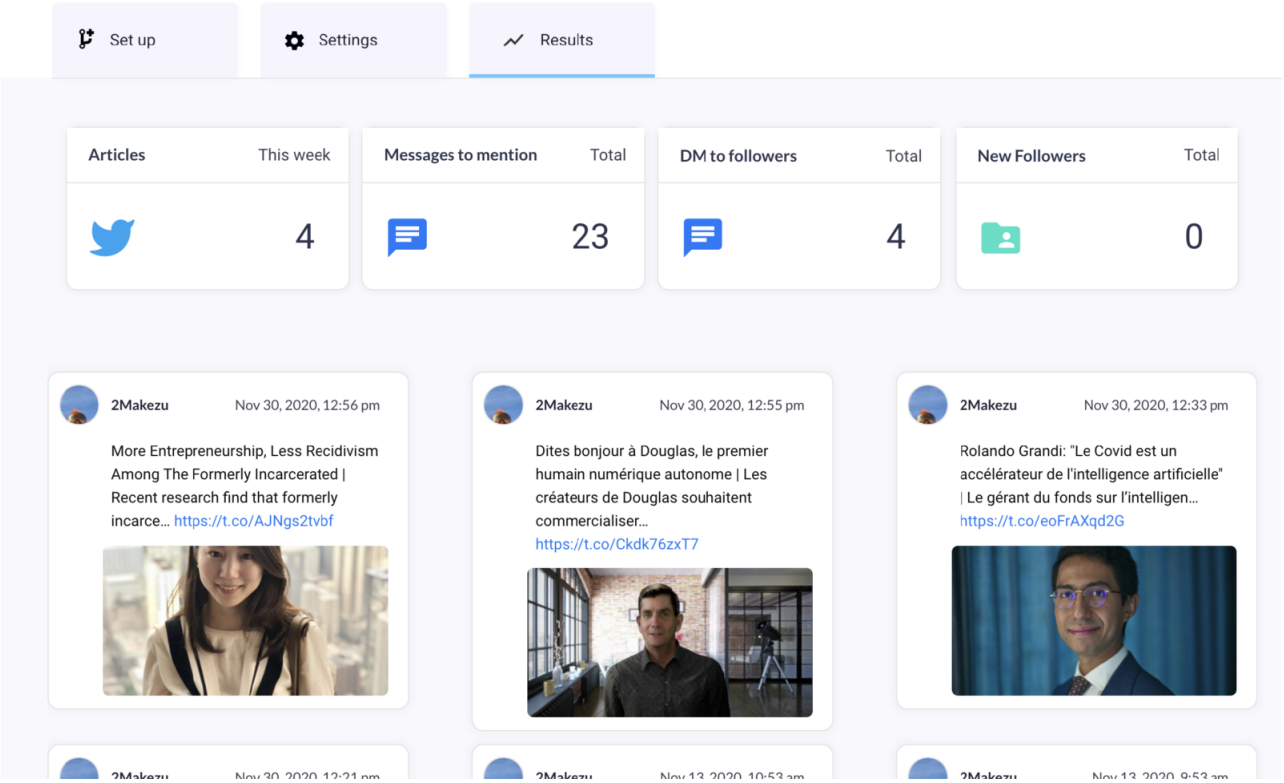


FIGURE 4.7: Interface utilisateur de la gestion du partage des articles

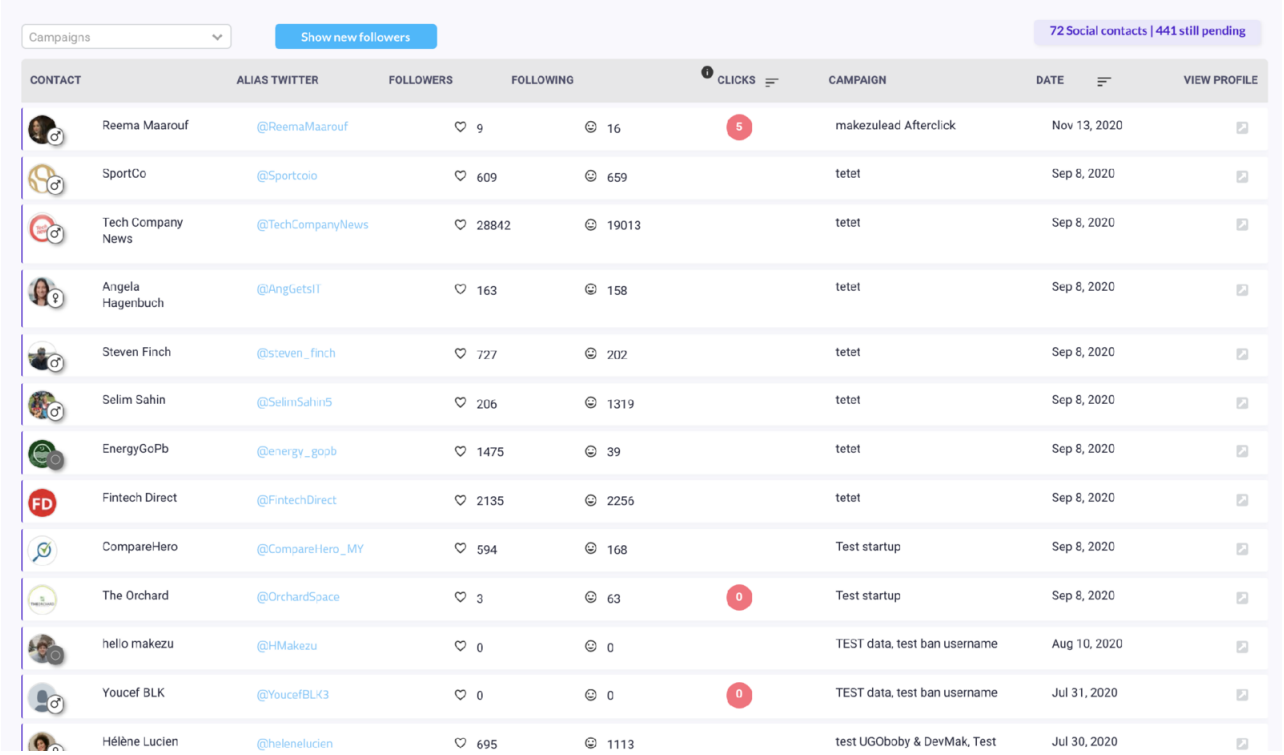


FIGURE 4.8: Gestion des informations des utilisateurs de Twitter

	authors	date_publish	description	maintext	title	url
0	['Philippe Escande', 'Philippe Van Parijs', 'L...	2020-09-29 07:00:16	Avec la crise sanitaire, des centaines de mill...	Manifestation a l'appel de la CGT, de Solidair...	" Mon secteur d'activité est mort " : le diffi...	https://www.lemonde.fr/economie/article/2020/0...
1	['Philippe Escande', 'Philippe Van Parijs', 'L...	2020-09-26 00:48:28	Séances de cinéma, résultats de football, mété...	DécryptagesSéances de cinéma, résultats de foo...	Sur Google, l'heure des recherches à " zéro cl...	https://www.lemonde.fr/economie/article/2020/0...
2	['Philippe Escande', 'Philippe Van Parijs', 'L...	2020-09-28 11:32:56	CHRONIQUE. La fraude montée par la propriétair...	Monique Piffaut, propriétaire des cassoulets W...	Scandale William Saurin : " A chaque affaire, ...	https://www.lemonde.fr/economie/article/2020/0...
3	['Philippe Escande', 'Philippe Van Parijs', 'L...	2020-09-28 05:45:08	TRIBUNE. Des responsables de la plate-forme ci...	Tribune. En pleine crise sociale, économique e...	" Face aux urgences sociales et environnementa...	https://www.lemonde.fr/idees/article/2020/09/2...
4	['Philippe Escande', 'Philippe Van Parijs', 'L...	2019-08-08 18:06:27	Vidéo - Le déploiement d'un réseau 5G national...	« Une catastrophe sanitaire en devenir. » C'es...	La 5G est-elle dangereuse pour votre santé ?	https://www.lemonde.fr/les-decodeurs/video/201...
...	...	...	...	...	...	...
211870	['Laurine Poret-Birembaux']	2021-01-20 11:00:07	Meghan Markle : Le père de Meghan Markle a fai...	Le père de Meghan Markle a fait de nouvelles c...	Meghan Markle en guerre avec son père, il fait...	https://www.melty.fr/meghan-markle-en-guerre-a...
211871	['Priscilla Brégeon-Minos']	2021-01-26 11:00:07	Chris Evans : Qui a les plus belles fesses des...	Qui a les plus belles fesses des Etats-Unis en...	Chris Evans (Avengers Endgame) a-t-il les fess...	https://www.melty.fr/chris-evans-avengers-endg...
211872	['Manon Roche']	2021-01-27 17:30:06	Épanouissement personnel : Comment ça, on est ...	Comment ça, on est en 2021 et tu ne manifestes...	Qu'est-ce que le manifesting, la nouvelle tend...	https://www.peaches.fr/qu-est-ce-que-le-manife...
211873	['Sébastien']	2021-01-22 18:00:09	Alix : Alix, qui est actuellement dans Les Pri...	Alix, qui est actuellement dans Les Princes et...	Alix (LPDLA8) au casting des Apprentis Aventur...	https://www.melty.fr/alix-lpdla8-au-casting-de...
211874	[]	2021-01-26 18:00:12	Huawei Mate 20 Pro : C'est une énorme chute po...	C'est une énorme chute pour un smartphone haut...	Bon Plan Huawei Mate 20 Pro : Pour les soldes,...	https://www.melty.fr/bon-plan-huawei-mate-20-p...

211875 rows × 6 columns

FIGURE 4.9: Affichage tronqué du fichier contenant les articles récupérés

Table des matières

Résumé technique **3**

1 Présentation de Makezu **4**
1.1 Présentation de l'entreprise 4
1.1.1 Brève histoire de l'entreprise 4
1.1.2 Makezu et les campagnes 4
1.1.3 Modèle économique 5
1.1.4 Concurrence et tendance 6
1.2 Présentation de l'équipe 6

2 Amélioration et évolution des algorithmes Makezu **8**
2.1 Amélioration de l'application et évolution des algorithmes d'intelligence artificielle . 8
2.2 Contributions 9
2.3 Environnement technologique 9
2.3.1 Bubble et AWS 9
2.3.2 Google Colab pour l'apprentissage automatique 10
2.4 Planning 10
2.4.1 Incubateur Station F 10
2.4.2 L'Oréal et Founders Factory 10
2.4.3 Lancement sur AppSumo 11
2.5 Prise de recul 11

3 Extraction et automatisation du contenu sur les différentes plateformes sociales **13**
3.1 Quora 13
3.1.1 Présentation de Quora et des outils utilisés 13
3.1.2 Méthodologie et fonctionnalités principales 13
3.1.3 Possibles améliorations du programme de récupération des informations . . . 14
3.2 Twitter 16
3.2.1 Analyse des tendances 16
3.2.2 Partage d'articles sur Twitter 18
3.2.3 Partage des questions et réponses Quora 19

3.2.4 Analyse du nom d'utilisateur pour déterminer le genre 20
3.2.5 Analyse de la biographie pour déterminer le genre 21
3.3 LinkedIn 22

4 Catégorisation des textes pour l'analyse des tendances et le profilage des utilisateurs sur Twitter 23
4.1 Introduction 23
4.1.1 Intérêts pour Makezu 23
4.1.2 Éléments d'histoire des modèles profonds de langage 23
4.2 Modèle de classification des textes en anglais 25
4.2.1 Ensemble de données du HuffPost 25
4.2.2 Présentation de BERT et RoBERTa 26
4.2.3 Affinement du modèle BERT 26
4.3 Élaboration d'un jeu de données composé d'articles francophones de journaux en ligne 29
4.3.1 Programme de récupération des articles 29
4.3.2 Annotation des articles 30
4.3.3 Pistes d'amélioration 31
4.4 Modèle de classification des textes en français 33
4.4.1 Présentation de FlauBERT 33
4.4.2 Affinement du modèle FlauBERT 33
4.4.3 Présentation du modèle CamemBERT 35
4.4.4 Affinement d'un modèle CamemBERT 35
4.5 Mise en production des modèles pour Makezu 35
4.5.1 Quantification des modèles 35
4.5.2 Distillation des modèles 36

Conclusion 38

Acronymes 46

Glossaire 47

Table des figures

1.1 Préparation d'une campagne sur `app.makezu.io` 5

3.1 POS tagging pour la phrase : *I like to play football. I hated it in my childhood though* 17
3.2 Représentation des structures latentes avec GloVe 18
3.3 Partage automatique d'articles sur le sujet « bitcoin » 19
3.4 Exemple de discussion autour du sujet Quora *Books* 20
3.5 Profil d'un utilisateur Twitter accompagné de sa biographie 21
3.6 Exemple de contenu partagé sur le fil d'actualité LinkedIn 22

4.1 Encodage du mot *it* dans un Transformer. Le mécanisme d'attention se concentre principalement sur *The Animal* 24
4.2 Distribution des catégories pour les articles du HuffPost récoltés 25
4.3 Cinq couches de *dropout* avec 50% de chance qu'un neurone ne soit pas activé. Chaque couche est suivie d'une couche entièrement connectée (*fully connected layer*) 27
4.4 Sortie tronquée du programme qui récupère les articles 30
4.5 Sortie tronquée des catégories les plus représentées 31
4.6 Quantification des poids durant l'entraînement 36
4.7 Interface utilisateur de la gestion du partage des articles 40
4.8 Gestion des informations des utilisateurs de Twitter 41
4.9 Affichage tronqué du fichier contenant les articles récupérés 41

Liste des tableaux

4.1 Hyperparamètres du modèle RoBERTa 28
4.2 Le nombre d'occurrence pour chaque classe 31
4.3 Poids assigné pour chaque classe 34
4.4 Hyperparamètres du modèle FlauBERT 34

Acronymes

API *Application Programming Interface*. 6, 8, 13, 16, 22, 29
AWS *Amazon Web Services*. 9

BERT *Bidirectional Encoder Representations from Transformers*. 24, 26, 28, 33, 34, 36, 38, 43

CPU *Central Processing Unit*. 10
CSV *Comma-separated values*. 15, 29, 30

GloVe *Global Vectors for Word Representation*. 17
GPU *Graphics Processing Unit*. 10, 33–35, 38

HTML *Hypertext Markup Language*. 13, 18
HTTP *Hypertext Transfer Protocol*. 18, 22

JSON *JavaScript Object Notation*. 22

LRAP *Label ranking average precision*. 34, 35

MLM *Masked Language Model*. 26

NER *Name Entity Recognition*. 16
NLP *Natural Language Processing*. 3, 24, 33, 38
NSP *Next Sentence Prediction*. 26

POS *Part-Of-Speech tagging*. 8, 16

RNN Réseaux de Neurones Récurrents. 24
RoBERTa *A Robustly Optimized BERT Pretraining Approach*. 24, 26, 35, 36, 43

SD *Syntactic Dependency Parsing*. 8

TPU *Tensor Processing Unit*. 10, 27, 28

WOEID *Where On Earth IDentifier*. 16

XML *Extensible Markup Language*. 13

Glossaire

apprentissage automatique Ensemble d'approches statistiques donnant aux ordinateurs la capacité d'≪ apprendre ≫ à partir de données. 4, 6, 10, 14, 26, 31, 47

apprentissage profond Ensemble de méthodes d'apprentissage automatique qui tente de modéliser des données grâce à des transformations non linéaires. 10, 17, 23, 26

follower Personne suivant le compte d'une autre personne sur Twitter. 6

roadmap Feuille de route permettant de communiquer et partager une intention stratégique pour atteindre des objectifs. 7

tokenisation Segmentation d'une phrase en plusieurs mots. 16

tweet Micromessage sur Twitter limité à 280 caractères. 4–6, 8, 16, 17, 19, 21, 23, 25, 29, 33, 35, 36

Bibliographie

[1] Jeffrey Pennington, Richard Socher, and Christopher D Manning. *Glove : Global Vectors for Word Representation.* In *EMNLP*, volume 14, pages 1532–1543, 2014.

[2] Julio Raffo and Gema Lax-Martinez. WGND 1.0, 2018.

[3] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. *Efficient Estimation of Word Representations in Vector Space*, 2013.

[4] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. *Language Models are Unsupervised Multitask Learners.* 2019.

[5] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. *XLNet : Generalized Autoregressive Pretraining for Language Understanding*, 2020.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. *BERT : Pre-training of Deep Bidirectional Transformers for Language Understanding*, 2018.

[7] 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*, 2019.

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. *Attention Is All You Need*, 2017.

[9] Rishabh Misra. *News Category Dataset*, 2018.

[10] Hiroshi Inoue. *Multi-Sample Dropout for Accelerated Training and Better Generalization*, 2020.

[11] Daniel Cer, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. *Universal Sentence Encoder*, 2018.

[12] Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoît Crabbé, Laurent Besacier, and Didier Schwab. *FlauBERT : Unsupervised Language Model Pre-training for French*, 2020.

[13] Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. *CamemBERT : a Tasty French Language Model. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[14] Guillaume Lample and Alexis Conneau. *Cross-lingual Language Model Pretraining*, 2019.

[15] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. *Unsupervised Cross-lingual Representation Learning at Scale*, 2020.

[16] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. *DistilBERT, a distilled version of BERT : smaller, faster, cheaper and lighter*. In *NeurIPS EMC Workshop*, 2019.

[17] Ben Hutchinson, Vinodkumar Prabhakaran, Emily Denton, Kellie Webster, Yu Zhong, and Stephen Denuyl. Social biases in nlp models as barriers for persons with disabilities, 2020.