RÉPUBLIQUE DU SÉNÉGAL

Un peuple - un but - une foi

¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR, DE LA RECHERCHE ET DE L'INNOVATON

¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤

Foo Nekk Foofu La

¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤

POLE SCIENCE, TECHNOLOGIES ET NUMERIQUE
MASTER : BIG DATA ANALYTICS

¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤

Sujet :

La modération automatique des commentaires toxiques dans les réseaux sociaux au Sénégal

¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤

Présenté par :

M. IBOU NGOM

M. MOHAMED KEBE

Sous la direction de :
Dr Edouard Ngor SARR
Sous la supervision de :
Prof Ousmane SALL

Membres du jury :
Président : Pr Alassane DIOP
Examinateur 1 : Youssou DIENG
Examinateur 2 : Khalifa SYLLA
Examinateur 3 : Seydina Moussa NDIAYE
Examinateur 4 : Gorgoumack SAMBE

Année universitaire : 2022 – 2023

Soutenu publiquement le 22/05/2025

Dédicace

Après avoir rendu grâce à ALLAH, le Tout-Puissant, nous dédions ce mémoire à nos chères mères et complices, qui sont toujours présentes à nos côtés et croient en nous.

Maman, nous vous adorons.

Remerciements

Nous adressons nos plus profonds et chaleureux remerciements à :

- Notre directeur de mémoire, Dr Edouard Ngor SARR pour nous avoir confié ce travail, pour le suivi, sa disponibilité, ses orientations et ses conseils ;
- Le Professeur Ousmane SALL pour le suivi et pour avoir accepté de supervisé notre mémoire ;
- Aux membres du jury pour avoir accepté d'évaluer ce travail ;
- L'ensemble des professeurs de l'Université Numérique Cheikh Hamidou Kane dont l'enseignement et les conseils ont été une source d'inspiration tout au long de nos études ;
- Au corps professoral de l'université pour sa disponibilité ;
- Nos pères et amis pour avoir toujours cru en nous ;
- Nos frères et sœurs pour leur soutien ;
- Toutes nos familles : nos tantes, nos oncles, nos cousins et cousines…pour leur accompagnement ;
- Tous les étudiants de la promotion 3 d'Atos plus particulièrement ceux de BDA ;
- Tous ceux qui, de près ou de loin, ont contribué à la rédaction de ce mémoire, selon leur degré de participation.

Résumé

Ce mémoire traite de la problématique de la détection et de la modération des commentaires toxiques sur internet, avec un focus particulier sur le contexte sénégalais. L'étude explore l'utilisation du Machine Learning pour identifier, classifier et modérer ces commentaires afin de préserver la liberté d'expression tout en garantissant un espace numérique respectueux et sécurisé. La recherche se divise en quatre parties principales : La clarification des concepts clés du sujet, l'état de l'art et positionnement, l'extraction des commentaires et la classification des commentaires toxiques. Une attention particulière est portée sur l'adaptation des algorithmes au contexte linguistique et culturel sénégalais, où le mélange des langues et des expressions constitue un défi particulier. Les résultats de l'étude montrent que l'utilisation combinée de différentes techniques de Machine Learning permet d'améliorer la détection des commentaires toxiques, contribuant ainsi à une modération plus efficace et plus précise.

Abstract

This thesis addresses the issue of detecting and moderating toxic comments on the internet, with a particular focus on the Senegalese context. The study explores the use of Machine Learning to identify, classify, and moderate these comments in order to preserve freedom of expression while ensuring a respectful and secure digital space. The research is divided into four main parts: clarification of the key concepts of the topic, state of the art and positioning, comment extraction, and toxic comment classification. Special attention is given to adapting algorithms to the linguistic and cultural context of Senegal, where the mixing of languages and expressions presents a particular challenge. The results of the study show that the combined use of different Machine Learning techniques improves the detection of toxic comments, thereby contributing to more effective and precise moderation.

Sommaire

Liste des figures

Liste des tableaux

Sigles et abréviation

- Sigle : Définition
- Hootsuite : Outil de gestion des réseaux sociaux.
- We Are Social : Agence de stratégie numérique et de marketing.
- PTN : Abbréviation de "putain", terme vulgaire informel.
- B1N : Abbréviation de "bien", terme informel.
- AI : Artificial Intelligence - Intelligence artificielle.
- ML : Machine Learning - Apprentissage automatique.
- Wolof : Langue locale du Sénégal.
- Sérère : Langue locale du Sénégal.
- Xaliss : Terme local pour "argent" au Sénégal.
- Deep Learning : Apprentissage profond, sous-catégorie du Machine Learning.
- NN : Neural Networks - Réseaux neuronaux.
- SVM : Support Vector Machine - Machine à vecteurs de support.
- Naive Bayes : Modèle basé sur le théorème de Bayes.
- K-means: Algorithme de clustering.
- Word2Vec : Modèle d'apprentissage pour générer des représentations vectorielles de mots.
- LSTM: Long Short-Term Memory - Mémoire à long terme.
- GRU : Gated Recurrent Unit - Unité récurrente avec porte.
- CNN : Convolutional Neural Network - Réseau neuronal convolutionnel.
- RNN : Recurrent Neural Network - Réseau neuronal récurrent.
- BERT : Bidirectional Encoder Representations from Transformers - Représentations d'encodeur bidirectionnel à partir des transformateurs.
- RoBERTa : A Robustly Optimized BERT Pretraining Approach - Une approche de pré-entraînement BERT optimisée de manière robuste.
- DistilBERT : Version allégée de BERT.
- ALBERT : A Lite BERT - Version allégée de BERT.
- T5 : Text-to-Text Transfer Transformer - Transformer de transfert texte-à-texte.
- API : Application Programming Interface - Interface de programmation permettant aux applications de communiquer.
- YouTube Data API v3 : API de YouTube permettant d'accéder aux données publiques de la plateforme.
- TFM : Télévision Futurs Médias - Chaîne de télévision sénégalaise.

- SEN TV : Sénégal Television - Chaîne de télévision sénégalaise.
- CSV : Comma Separated Values - Format de fichier texte pour stocker des données tabulaires.
- VSCode : Visual Studio Code - Éditeur de code source développé par Microsoft.
- Jupyter Notebook : Environnement interactif pour créer et partager des documents de code et d'explications.
- Google Cloud Console : Interface de gestion des services cloud de Google.
- Google Sheets : Outil de feuille de calcul en ligne de Google.
- Google Drive : Service de stockage en ligne de Google.
- Anaconda : Distribution Python pour la gestion des environnements virtuels et des dépendances.
- Pandas : Bibliothèque Python pour manipuler des données sous forme de tableaux (DataFrame).
- NumPy : Bibliothèque Python pour le calcul scientifique.
- Scikit-learn : Bibliothèque Python pour les tâches d'apprentissage automatique.
- TensorFlow : Framework open-source développé par Google pour l'apprentissage automatique.
- Torch : Bibliothèque de machine learning, principalement utilisée pour le deep learning.
- GPU : Graphics Processing Unit - Unité de traitement graphique utilisée pour l'entraînement des modèles.
- ID : Identifiant unique attribué à chaque vidéo ou événement de diffusion.
- Re (Regex) : Regular Expression - Expression régulière utilisée pour manipuler des chaînes de caractères.
- FC : Fully Connected - Couche entièrement connectée dans un réseau neuronal.
- Adam : Adaptive Moment Estimation - Optimiseur utilisé en apprentissage automatique.
- CrossEntropyLoss : Perte d'entropie croisée, fonction de perte couramment utilisée pour les tâches de classification.
- K-Fold : Validation croisée à K plis, méthode d'évaluation des modèles.
- Stratified K-Fold : Validation croisée stratifiée à K plis, une variante du K-Fold où chaque pli conserve les proportions des classes d'origine.
- Dropout : Méthode de régularisation utilisée pour éviter le surapprentissage.
- Embedding : Représentation vectorielle des mots, technique pour transformer les mots en vecteurs numériques.
- Accuracy : Précision, proportion de prédictions correctes du modèle.

- Training Loss : Perte d'entraînement, fonction de coût utilisée pendant l'entraînement.
- Validation Loss : Perte de validation, fonction de coût calculée sur le jeu de validation.
- Training Accuracy : Précision d'entraînement, proportion de prédictions correctes sur l'ensemble d'entraînement.
- Validation Accuracy : Précision de validation, proportion de prédictions correctes sur l'ensemble de validation.

Introduction Générale

Naturellement, l'humain ressent le besoin de communiquer, partager, débattre et échanger, parfois de manière animée [1]. En effet avec l'avènement des nouvelles technologies et d'Internet, nous assistons à la naissance de modes de communications innovants, tels que les réseaux sociaux, les forums et les blogs. Aujourd'hui, Internet domine la diffusion d'informations à l'échelle mondiale et s'impose comme un outil de communication incontournable. Il offre à chaque utilisateur une liberté d'expression inédite, permettant de partager des opinions, des expériences et des idées sur divers sujets. Selon un rapport publié en 2021 par NOISY DIGITAL, Hootsuite et We Are Social, le Sénégal recense plus de 7,81 millions d'internautes, dont 3,90 millions sont actifs sur les médias sociaux numériques[1]. Ces plateformes, grâce à leurs fonctionnalités interactives, permettent aux utilisateurs non seulement de s'informer, mais aussi de partager leurs émotions, notamment à travers les sections de commentaires.

Les réseaux sociaux numériques occupent ainsi une place centrale dans nos vies, devenant des espaces privilégiés où chacun peut mettre en avant certains aspects de sa personnalité, créer un profil unique et interagir avec diverses communautés.

Mais face à cette tendance se pose le problème de propos toxique dans les espaces commentaires. Ce qui remet en cause la sécurité et le bien être des utilisateurs d'internet. **Ainsi, comment l'usage du Machine Learning peut-il nous aider à améliorer la détection et la modération des propos toxiques sur internet ?** En d'autres termes :

- Les algorithmes de Machine Learning peuvent-ils identifier la toxicité des commentaires sur internet ?
- Pouvons-nous aspirer à de meilleures modération afin d'anticiper la publication des commentaires toxiques ?

Face à cette problématique, nous proposons de mettre en place un système de modération capable de :

- Détecter automatiquement les commentaires toxiques ;
- Supprimer si nécessaire les commentaires détectés ;
- Préserver la liberté d'expression.

[1] https://noisydigital.com/fr_fr/les-chiffres-du-numerique-en-2021-au-senegal/#:~:text=Les%20statistiques%20d%C3%A9montrent%20qu'il,23%25%20de%20la%20population%20totale.&text=3.4%20millions%20de%20personnes%20se,nombre%20total%20d'utilisateurs%20actifs.

Notre objectif principal est d'explorer les multiples aspects liés à la modération des propos toxique par machine learning sur internet, en examinant à la fois les bases théoriques, les avancées technologiques et les applications déjà expérimentées. En d'autres termes, l'objectif est d'utiliser des techniques de machine learning pour examiner le fonctionnement de la modération des propos toxiques dans les commentaires et évaluer son efficacité dans les sites internet au Sénégal. Pour atteindre cet objectif général, nous nous focaliserons sur trois objectifs spécifiques à savoir :

- Analyser les concepts et méthodes de base de la modération des commentaires toxiques par machine learning ;
- Etudier les avancées récentes en passant par la revue de la littérature existante ;
- Illustrer l'application pratique du machine learning dans la modération des commentaires toxiques avec une étude de cas concrète.

Cette étude à un double intérêt à la fois pour les organes étatiques de contrôle et pour les organes médiatiques. Ainsi, à terme, cette étude permet de :

- Préserver un espace d'expression libre, respectueux et **v**aloriser ;
- Soutenir la régulation et la gouvernance des contenus en ligne ;
- Renforcer la confiance des citoyens dans l'espace numérique.

Nous formulons les hypothèses suivantes pour orienter notre recherche :

- L'adaptation d'algorithmes de machine Learning au contexte linguistique et culturel sénégalais peut renforcer l'efficacité et la précision dans la détection des commentaires toxiques ;
- La combinaison de techniques de machine Learning peut améliorer significativement la modération.

Ces hypothèses guident l'analyse approfondie de la modération des commentaires toxiques sur internet dans le contexte sénégalais.

Afin de bien mener cette étude, nous utiliserons une approche qualitative. En effet, L'approche qualitative se concentre sur l'analyse des phénomènes en profondeur à travers l'interprétation des données. Elle vise à comprendre les significations, les perceptions et les contextes associés à un sujet donné. Dans le cadre de la modération des commentaires toxiques dans un pays où le langage est souvent mixte, une approche qualitative est particulièrement pertinente pour les raisons suivantes :

- **La Complexité linguistique et culturelle :** l'expression utilisée au Sénégal est composée à 80% de la langue Wolof[2], en raison de son absence de standardisation et de sa richesse culturelle, elle contient des nuances et des expressions idiomatiques qui ne peuvent pas être pleinement comprises ou interprétées par des méthodes purement quantitatives. L'approche qualitative permet d'explorer ces subtilités linguistiques et contextuelles ;
- **L'Analyse des contextes d'usage** : Les commentaires toxiques ne sont pas toujours explicites. Parfois, leur toxicité dépend du ton, de l'intention ou du contexte. Une analyse qualitative permet d'identifier ces éléments implicites qui pourraient échapper à une approche basée uniquement sur des chiffres.

Notre étude est divisée en quatre grands chapitres qui sont :

- **Chapitre 1 : Clarification des concepts clés.** Dans ce chapitre, nous abordons les généralités sur les commentaires, en expliquant leur rôle dans les interactions numériques et les leurs impacts. Il introduit également les notions de Machine Learning, en mettant en lumière son importance pour analyser et traiter les données ;
- **Chapitre 2 : État de l'art et positionnement.** Dans ce chapitre nous traitons l'état de l'art et du positionnement de l'étude. Il présente les travaux précédents sur la modération des commentaires toxiques, les approches et outils existants. Ce chapitre permet de situer notre contribution en identifiant les défis et en justifiant l'approche retenue pour y répondre ;
- **Chapitre 3 : Extraction des commentaires.** Ce chapitre explique comment les données nécessaires ont été collectées pour l'étude. Il décrit les plateformes ciblées (comme les réseaux sociaux), les méthodes utilisées pour extraire les commentaires, et le traitement appliqué aux données à savoir le nettoyage et l'étiquetage ;
- **Chapitre 4 : Classification des commentaires toxiques.** Et enfin nous expliquons dans ce dernier chapitre comment le système de détection a été conçu et testé. Il explique les algorithmes utilisés, leur entraînement sur les données collectées, et l'évaluation de leurs performances. Il détaille également les ajustements faits pour gérer les langues mixtes et les spécificités culturelles. Enfin, il montre les résultats obtenus et comment le modèle détecte les commentaires toxiques dans des conditions réelles.

[2] https://www.inalco.fr/sites/default/files/asset/document/fiche_wolof.pdf

Chapitres 1 : Clarification des concepts clés du sujet

1.1. Généralité sur les commentaires et problématique

1.1.1. Commentaires

Un commentaire[3] est une expression d'opinion ou une remarque visant à interpréter une situation, souvent de manière concise. Il peut servir à poser une question, formuler une critique, ou compléter une information sur un sujet précis. Sur internet, cela correspond à des messages que les utilisateurs partagent sur des plateformes comme les réseaux sociaux, les forums ou les blogs.

1.1.2. Commentaires toxiques

Les commentaires toxiques en ligne désignent des messages marqués par des comportements négatifs, hostiles ou nuisibles envers des individus, des groupes ou des opinions. Ils ne se limitent pas aux attaques directes, mais englobent aussi des actions visant à susciter des réactions émotionnelles intenses ou à exacerber les tensions par des propos provocateurs. Ce type de contenu crée un climat néfaste, pouvant avoir des répercussions psychologiques graves sur les victimes, telles que le stress, l'anxiété ou même la dépression. Par ailleurs, la toxicité en ligne altère la qualité des discussions et perturbe l'équilibre des communautés, compliquant ainsi le partage d'idées constructives et enrichissantes.

1.1.3. Typologie des commentaires toxiques

Les commentaires toxiques peuvent se manifester sous différentes formes, à savoir :

- **Les insultes :** ce sont des expressions de tensions verbales qui surgissent dans des situations de communication impliquant au moins deux individus. L'acte d'insulter résulte de la combinaison de plusieurs éléments, notamment le contexte de l'énonciation, la langue utilisée et les enjeux sous-jacents à la communication [2] ;
- **Le harcèlement** : les technologies numériques favorisent parfois des comportements à risque, tels que les rencontres avec des inconnus, la diffusion de contenus intimes ou l'exposition à des sites inappropriés. Le harcèlement en ligne, est devenu l'une des formes les plus répandues de violence, incluant parfois des dimensions sexuelles. Il est essentiel d'explorer les enjeux liés à cette problématique pour mieux encadrer les pratiques et apporter des repères clairs dans un cadre professionnel [3] ;
- **La diffamation** : elle se concentre sur des déclarations ou publications qui portent atteinte à la réputation et à la dignité des individus concernés. La diffamation est particulièrement préoccupante dans le contexte numérique, surtout lorsqu'il s'agit de contenu généré par les utilisateurs, tels que les publications sur les réseaux sociaux, les

[3] https://www.cours-thales.fr/lycee/premiere/le-commentaire-litteraire

blogs ou les forums. Ce type de contenu peut se propager rapidement et toucher un large public, amplifiant ainsi l'impact négatif sur la personne visée. Internet, grâce à son interactivité et sa capacité à diffuser de l'information à une échelle mondiale, pose des défis uniques qui distinguent ce médium des formes traditionnelles de communication, telles que la presse écrite ou audiovisuelle [4] ;

- **Le discours de haine** : c'est souvent une attaque envers des groupes entiers, favorisant la discrimination et la marginalisation. Bien que les propos discriminatoires ou offensants aient toujours existé, l'expression "discours de haine" est relativement récente et trouve son origine légale en France avec la loi Gayssot, qui condamne les propos racistes. Avec l'essor des réseaux sociaux et des forums en ligne, ces discours hostiles ont gagné en intensité et en visibilité, se propageant rapidement et atteignant un public beaucoup plus large [5].

1.1.4. Les défis de la détection et de la modération des commentaires toxiques

Dans le contexte numérique actuel, les commentaires toxiques posent des défis critiques en matière de détection et de modération. Nous distinguons deux types de modération :

- **La modération proactive** : elle consiste à examiner tous les commentaires avant leur publication et décider s'ils sont acceptables [6]. Elle est un outil puissant, mais elle nécessite un équilibre entre automatisation et intervention humaine pour garantir son efficacité et son équité ;
- **La modération réactive** : elle consiste à autoriser la publication de messages sans être révisés à l'avance, et à agir après coup si un message est jugé toxique [6], souvent en utilisant un système de rapportage par les utilisateurs de la communauté. En effet la modération intervient après un signalement des utilisateurs grâce à des outils mis en leur disposition.

Cependant malgré les formes de modérations mis en place il existe plusieurs défis à relever et sur ceux on peut citer :

- **La difficulté à définir les seuils de toxicité** : la toxicité des commentaires est une notion subjective qui dépend du contexte, de la culture et même des dynamiques de chaque plateforme. La toxicité ne se limite pas seulement aux insultes, mais peut englober des sous-entendus ou des formes de violence psychologique plus subtiles. Il est complexe d'établir un seuil unique qui s'appliquerait uniformément à tous les types d'interactions en ligne. Par exemple, certains termes acceptés dans des conversations informelles peuvent être considérés comme toxiques lorsqu'ils sont employés dans un

autre contexte, rendant difficile la création de critères objectifs. Un aspect crucial dans l'analyse de la toxicité linguistique est le contexte culturel, qui peut considérablement influencer la perception de certains termes ou expressions. Au Sénégal, par exemple, les différences régionales et culturelles rendent certains propos plus sensibles ou incompris lorsqu'ils sont interprétés en dehors de leur cadre d'origine. Une illustration pertinente de ce phénomène est l'expression « diokhal ma sa ndeye », couramment employée dans les campagnes pour signifier « donne à ta mère » et exprimer une demande de manière familière. Dans ce contexte rural, cette expression n'a aucune connotation offensante ; elle s'inscrit au contraire dans une tradition de solidarité et de respect pour la famille. Cependant, cette même expression peut être perçue différemment dans des milieux plus urbains ou auprès de personnes moins habituées à ce type de langage. Dans les villes, et surtout dans les milieux où l'usage du français ou d'autres formes plus formelles de communication prédomine, le terme « ndeye » (mère) peut être perçu comme intrusif, voir irrespectueux. Ce décalage de perception souligne l'importance de prendre en compte les spécificités culturelles dans l'établissement de seuils de toxicité dans un système de modération automatique. La détection de toxicité doit ainsi aller au-delà de la simple détection lexicale ; elle doit être capable de reconnaître les nuances culturelles et contextuelles propres à chaque communauté linguistique, sous peine d'aboutir à une modération inadaptée et injuste. La construction de tels modèles nécessite donc une compréhension approfondie des contextes d'utilisation afin de minimiser les erreurs de jugement et d'améliorer l'efficacité de la modération des contenus en ligne. L'autre cas contextuel qu'on peut donner c'est le cousinage à plaisanterie qui est profondément enraciné dans les relations sociales quotidiennes. Par exemple, les Peuls et les Sérères s'adressent souvent des taquineries telles que *"Sama diam bi"* (mon esclave) ou *"Sérère dou nit"* (le Sérère n'est pas une humain). Bien que ces expressions puissent sembler offensantes en dehors de leur contexte, elles sont perçues localement comme des marques d'humour et de complicité qui renforcent le vivre-ensemble et les liens communautaires. Cependant, dans un système de modération automatisée des commentaires, ces expressions risquent d'être interprétées à tort comme toxiques, mettant ainsi en lumière la complexité culturelle et linguistique du contexte sénégalais ;

- **Un langage mixte ou *code*-switching [4]:** La création d'un corpus multilingue adapté au contexte sénégalais constitue un défi de taille pour la détection de la toxicité dans les commentaires en ligne. Le Sénégal se caractérise par une grande diversité linguistique, avec une utilisation fréquente du wolof, du sérère, du français, de l'anglais (…) dans les échanges numériques. Cette pratique, connue sous le nom de langage mixte ou code-switching, complique considérablement l'analyse automatisée. Les modèles de machine Learning souvent conçus pour des langues monolingues ou standardisées, échouent à interpréter correctement des phrases où plusieurs langues coexistent. Par exemple, une phrase comme *"Boy temps yii yagui lekk sa xaliss"* (Tu as beaucoup d'argent frère ces temps-ci) illustre parfaitement cette difficulté ;
- **L'absence de ressources linguistiques standardisées :** Pour le wolof ou d'autres langues locales nous constatons qu'il n'y a pas de gros dictionnaire de données standardisées, ce qui aggrave le problème. Contrairement au français ou à l'anglais [7], ces langues ne disposent pas de grands corpus annotés, de dictionnaires numériques exhaustifs ou de bases de données lexicales accessibles. Cela limite la capacité des modèles à reconnaître et interpréter ces langues, qu'il s'agisse de leur structure, de leur grammaire ou de leur vocabulaire ;
- **L'abréviation :** Nous constatons une utilisation massive de formes abrégées et de termes informels dans les discussions en ligne. Les internautes sénégalais emploient fréquemment des abréviations ou des expressions idiomatiques qui échappent aux systèmes de modération traditionnels. Par exemple, *"ptn"* pour "putain" ou *"b1n"* pour "bien" peuvent avoir une connotation toxique ou non, en fonction du contexte. Ces expressions ne sont pas seulement linguistiques, mais elles sont aussi ancrées dans les pratiques culturelles locales, ce qui rend leur détection d'autant plus complexe.

En résumé, la diversité linguistique, l'absence de ressources spécifiques, l'usage d'abréviations rendent la création d'un corpus multilingue riche, mais extrêmement complexe. Relever ce défi est une étape incontournable pour développer des systèmes efficaces de détection de la toxicité adaptés à la réalité de notre pays.

[4] https://www.laroutedeslangues.com/blog/code-switching-une-menace-ou-un-developpement-enrichissant/#:~:text=Le%20code%2Dswitching%20(l',fun%2C%20let's%20go%20!%20%C2%BB.

1.1.5. Le Machine Learning

1.1.5.1. Définitions

Le Machine Learning, ou apprentissage automatique, est une branche de l'intelligence artificielle qui vise à permettre aux systèmes informatiques de tirer des leçons à partir de données, sans intervention humaine directe[5]. Introduit pour la première fois par Arthur Samuel dans les années 1950, ce domaine est défini comme une approche permettant aux machines d'apprendre sans être explicitement programmées[6]. En d'autres termes, le Machine Learning offre aux ordinateurs la capacité d'acquérir de nouvelles compétences en analysant des données et en affinant leurs performances grâce à l'expérience.

Le concept repose sur l'utilisation d'algorithmes capables d'identifier des schémas ou relations au sein d'un ensemble de données. Ces algorithmes peuvent ensuite effectuer des prédictions ou prendre des décisions en fonction de ce qu'ils ont appris, tout en continuant à améliorer leur précision avec l'ajout de nouvelles données.

1.1.5.2. Fonctionnement

Le fonctionnement du Machine Learning peut être résumé comme un processus d'entraînement au cours duquel un algorithme analyse des données pour reconnaître des modèles. Il repose sur l'idée fondamentale qu'il existe une relation mathématique sous-jacente entre les données[7], permettant de les organiser, de les regrouper ou de les classer. Pour exploiter cette relation, il faut concevoir un algorithme adapté.

Cet algorithme permet à la machine d'apprendre progressivement à partir des données fournies, grâce à des traitements itératifs, jusqu'à atteindre un certain niveau d'autonomie dans ses prédictions ou décisions.

[5] https://www.elastic.co/fr/what-is/machine-learning
[6] https://blog.clevy.io/fr/introduction-machine-learning-1-3-histoire/#:~:text=En%201959%2C%20c'est%20l,son%20programme%20cr%C3%A9%C3%A9%20en%201952.
[7] https://www.jedha.co/formation-ia/formation-machine-learning

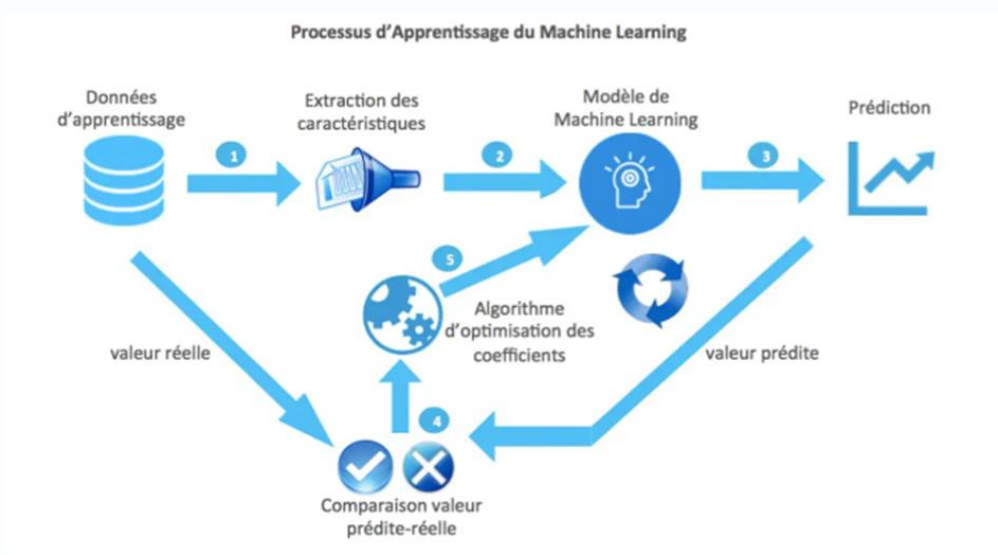

Figure 1: Principe de fonctionnement du machine Learning [8]

Comme sur la figure le fonctionnement du processus d'apprentissage commence par la collecte de données d'apprentissage, qui servent à entraîner le modèle. Ces données brutes sont ensuite transformées en caractéristiques pertinentes à travers un processus appelé extraction de caractéristiques, afin de rendre les informations utilisables par le modèle. Puis utilise ces caractéristiques pour générer des prédictions qui sont ensuite comparées aux valeurs réelles associées aux données d'entraînement pour mesurer leur exactitude. La différence entre les valeurs prédites et réelles (appelée erreur) est calculée et utilisée pour ajuster les paramètres internes du modèle grâce à un algorithme d'optimisation.

Ce processus est *itératif* : le modèle est constamment réévalué et amélioré jusqu'à ce que l'erreur soit suffisamment faible. Une fois le modèle optimisé, il peut être utilisé pour prédire des résultats sur de nouvelles données avec un niveau élevé de précision.

1.1.5.3. Phase de l'apprentissage machine

1.1.5.3.1. Collecte de données

La collecte de données de haute qualité est essentielle pour mener des études[9]. Elle est une démarche qui vise à identifier et interroger des sources d'informations pertinentes afin d'acquérir des données appropriées et capables de répondre aux objectifs fixés ou aux besoins spécifiques d'une recherche [10].

La première étape de tout projet de Machine Learning consiste donc à rassembler des données pertinentes. Elle commence par l'identification des informations nécessaires et des méthodes

[8] https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.jedha.co%2Fformation-ia%2Fformation-machine-learning&psig=AOvVaw0BHqUi0qHVnayI5aYkytCZ&ust=1732843830450000&source=images&cd=vfe&opi=89978449&ved=0CBcQjhxqFwoTCOjCh7rw_YkDFQAAAAAdAAAAABAE

[9] https://www.questionpro.com/blog/fr/collecte-de-donnees/

adéquates pour les obtenir. Elle permet également de tester des hypothèses à partir des données recueillies. En général, cette étape est cruciale et constitue un fondement essentiel de tout processus de recherche. Les méthodes de collecte varient en fonction du domaine d'étude et des objectifs visés, s'adaptant aux spécificités des informations recherchées.

1.1.5.3.2. Prétraitement de données

Une fois collectées, les données brutes doivent être transformées pour être utilisables par les algorithmes. Le prétraitement des données consiste à préparer les données brutes pour qu'elles soient utilisables par un modèle de Machine Learning [11]. Cette étape inclut l'élimination des valeurs manquantes, la normalisation des échelles et, parfois, l'annotation des données pour les rendre exploitables dans le cas des modèles supervisés. Une préparation soignée garantit que les algorithmes reçoivent des entrées optimales.

1.1.5.3.3. Sélection des caractéristiques

La sélection des caractéristiques consiste à identifier les variables les plus importantes qui influencent le problème à résoudre. Cette démarche vise à réduire la complexité du modèle, éviter les redondances ou les informations inutiles, et se concentrer sur les données les plus significatives pour la tâche à accomplir. Cela permet non seulement de gagner du temps et des ressources, mais aussi d'améliorer la précision et la capacité de généralisation du modèle sur de nouvelles données.

1.1.5.3.4. Sélection des modèles

Pour traiter les données, plusieurs modèles de Machine Learning peuvent être utilisés. Cependant il faut choisir le modèle d'apprentissage le plus approprié pour résoudre un problème spécifique. Cela implique de comparer plusieurs algorithmes ou configurations en fonction de leurs performances sur un ensemble de données, en tenant compte de critères tels que la précision, la vitesse, la robustesse ou la capacité de généralisation. L'objectif est de trouver le modèle qui offre le meilleur équilibre entre complexité et performance pour répondre aux exigences.

1.1.5.3.5. Entrainement

L'entraînement[10] en machine Learning est le processus par lequel un modèle apprend à effectuer une tâche spécifique à partir de données. Ce processus consiste à fournir au modèle des exemples, à comparer ses prédictions avec les résultats attendus, et à ajuster ses paramètres

[10] https://www.databricks.com/fr/glossary/machine-learning-models#:~:text=Qu'est%2Dce%20que%20l,ou%20certaines%20donn%C3%A9es%20de%20sortie.

pour minimiser les erreurs. L'objectif est que le modèle soit capable de bien généraliser et de fournir des prédictions précises sur des données qu'il n'a jamais vues.

1.1.5.3.6. Tests

Une fois le modèle entraîné, il sera évalué sur un ensemble de données séparé, appelé ensemble de test. Cela permet de mesurer sa capacité à généraliser sur de nouvelles données qu'il n'a jamais vues auparavant. Les résultats obtenus sur l'ensemble de test permettent de comparer les performances avec d'autres modèles et d'identifier celui qui convient le mieux.

1.1.5.3.7. Déploiement du modèle

Le modèle peut être déployé dans un environnement de production pour fournir des prédictions en temps réel. Cela implique souvent une intégration avec des systèmes existants pour automatiser les processus décisionnels.

1.1.5.3.8. Monitoring et mise à jour

Une fois en production, il est essentiel de surveiller les performances du modèle pour détecter toute dégradation éventuelle. Avec l'évolution des données ou des besoins, des mises à jour périodiques du modèle peuvent être nécessaires pour maintenir sa pertinence et sa précision.

1.1.5.4. Approches de machines Learning

1.1.5.4.1. Approche supervisée

L'apprentissage supervisé repose sur des ensembles de données annotées, où chaque exemple est associé à une étiquette ou une variable cible. L'objectif est de permettre au modèle de prédire correctement les résultats pour de nouvelles données. Ce type d'apprentissage repose sur un ensemble de données, où chaque entrée est associée à une réponse connue, facilitant ainsi l'apprentissage par exemple. Le modèle ajuste ses paramètres en fonction de l'écart entre ses prédictions et les valeurs attendues afin de minimiser l'erreur [12]. Ce type d'apprentissage est couramment utilisé pour des tâches telles que :

- **La classification [11]:** où l'on détermine à quelle catégorie appartient une donnée par exemple, distinguer un email spam d'un email légitime, classer un commentaire de toxique ou non toxique ;
- **La régression [12]:** où l'on prédit une valeur continue comme le prix d'une maison en fonction de ses caractéristiques.

[11] https://24pm.com/117-definitions/505-classification
[12] https://projeduc.github.io/intro_apprentissage_automatique/regression.html

La figure ci-dessous illustre un parfait exemple d'algorithme d'apprentissage supervisé où les données étiquetées, composées de formes associées à leurs catégories (cercle, triangle, rectangle, etc.), sont utilisées pour entraîner le modèle. Le modèle apprend à reconnaître les formes en analysant ces exemples. Une fois l'entraînement terminé, le modèle obtenu peut être utilisé pour faire des prédictions sur de nouvelles données.

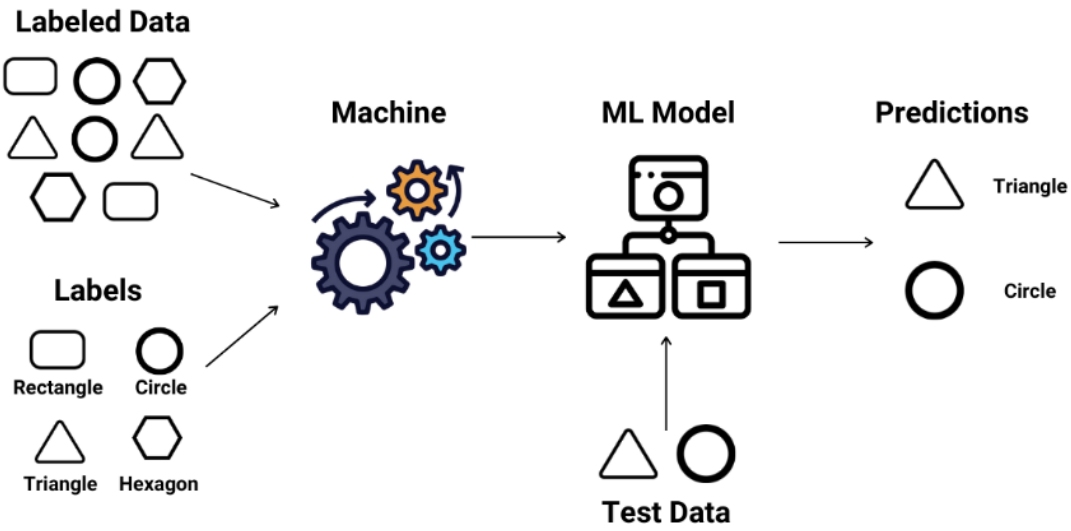

Figure 2 : Fonctionnement d'algorithme de classification supervisé [13]

1.1.5.4.2. Approche non supervisée

L'apprentissage non supervisé est une technique où le modèle travaille sur des données non étiquetées, sans connaître les réponses attendues. L'objectif est de découvrir des structures ou des modèles cachés dans les données. Contrairement à l'apprentissage supervisé, cette méthode n'a pas de sortie cible définie et se concentre sur l'identification des similitudes ou des différences au sein des données [12]. Cette approche est fréquemment utilisée pour :

- **Le clustering [14]:** où les données sont segmentées en groupes selon leurs caractéristiques communes ;
- **La réduction de dimensionnalité** [15]: qui vise à simplifier les données tout en conservant leur structure essentielle.

Une parfaite illustration est présentée sur la figure 3 suivante qui montre un ensemble de données (fruits) passer au modèle et ce dernier apprend à les caractériser selon leur ressemblance et donner le résultat final.

[13] https://blent.ai/blog/a/apprentissage-supervise-definition
[14] https://pythonds.linogaliana.fr/content/modelisation/5_clustering.html
[15] https://cedric.cnam.fr/vertigo/Cours/ml/coursReductionDimension.html

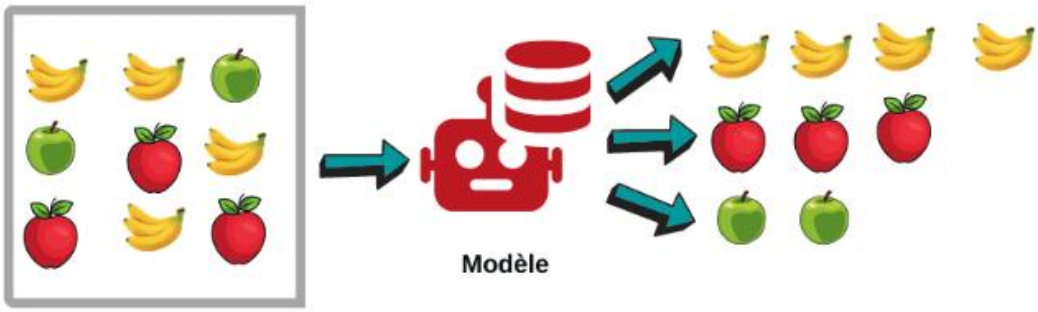

Figure 3 : Fonctionnement d'algorithme de classification non supervisé[16]

1.1.5.4.3. Approche semi supervisée

L'apprentissage semi-supervisé quant à elle, combine des données étiquetées et non étiquetées, ce qui en fait une approche hybride entre les deux précédentes. Cette méthode est particulièrement utile lorsque l'étiquetage des données est coûteux ou difficile à réaliser. Les données étiquetées fournissent un cadre pour guider le processus d'apprentissage, tandis que les données non étiquetées enrichissent l'ensemble d'entraînement, permettant au modèle de mieux généraliser [13].

Une parfaite illustration est présentée sur la figure 4 suivante, qui montre comment un algorithme d'apprentissage semi-supervisé apprend à reconnaître des objets. Il utilise des données d'entrée, certaines étiquetées (comme "Orange" ou "Banana") et d'autres non étiquetées (comme une pomme inconnue). Grâce à l'apprentissage, le modèle peut analyser les caractéristiques des données et prédire correctement, par exemple, que le fruit non étiqueté est une pomme.

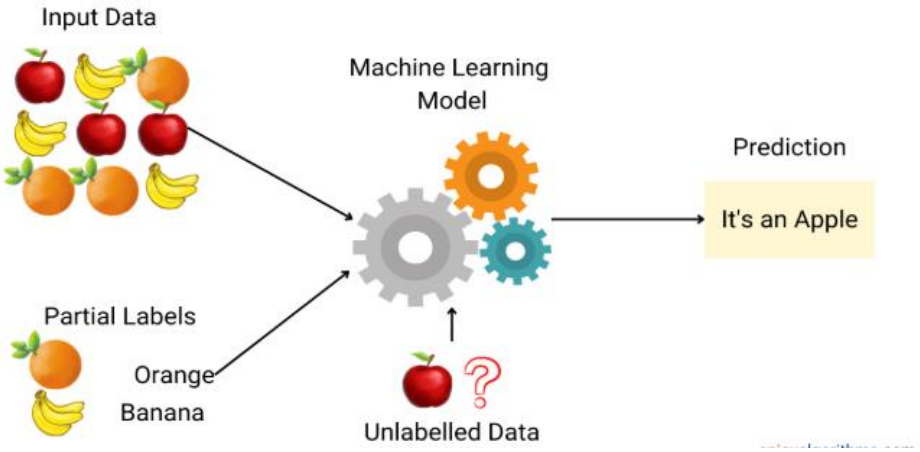

Figure 4 : Fonctionnement d'algorithme de classification semi-supervisé [17]

[16] https://phedone.com/fr/blog/artificial-intelligence-what-is-the-unsupervised-learning/
[17] https://github.com/0liverFlow/HandwritingRecognition

1.1.5.4.4. Approche par renforcement

L'apprentissage par renforcement repose sur un mécanisme d'essais et d'erreurs, où un agent apprend en interagissant avec son environnement. L'agent est récompensé pour les actions correctes et pénalisé pour les erreurs, ce qui l'incite à adopter des stratégies optimales. En interagissant avec son environnement, l'agent optimise ses actions pour maximiser une récompense cumulative (voir figure 5).

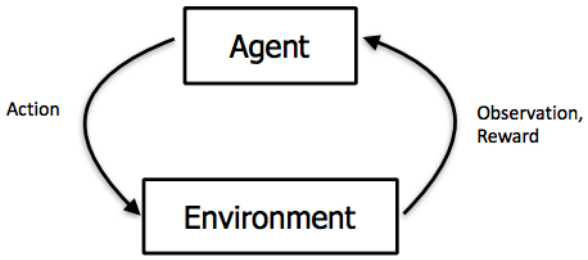

Figure 5 : Fonctionnement d'algorithme d'apprentissage par renforcement[18]

1.1.5.4.5. Approche par deep learning

Le deep learning, ou apprentissage profond, est une branche essentielle du machine learning. Il s'appuie sur des algorithmes capables de reproduire le fonctionnement du cerveau humain grâce à des réseaux de neurones artificiels. Ces réseaux sont formés de nombreuses couches, parfois des dizaines ou même des centaines, où chaque couche analyse et transmet les informations à la suivante. Il se distingue par sa capacité à extraire automatiquement des caractéristiques hiérarchiques à partir de grandes quantités de données, en allant des éléments simples aux structures plus abstraites. Cette approche est particulièrement efficace pour des tâches nécessitant une analyse détaillée et automatisée des données [14]. La notion de deep Learning peut être illustrée sur la figure 6 qui montre comment un réseau de neurones est utilisé pour classifier des images. Lors de l'apprentissage, le réseau reçoit des images étiquetées et ajuste ses connexions internes pour corriger les erreurs de classification. Après l'entraînement, il peut analyser de nouvelles images non étiquetées et produire des prédictions correctes, comme identifier un chat dans une image. Cela illustre le principe d'entraînement et d'application du deep Learning pour la reconnaissance d'images.

[18] https://larevueia.fr/apprentissage-par-renforcement/

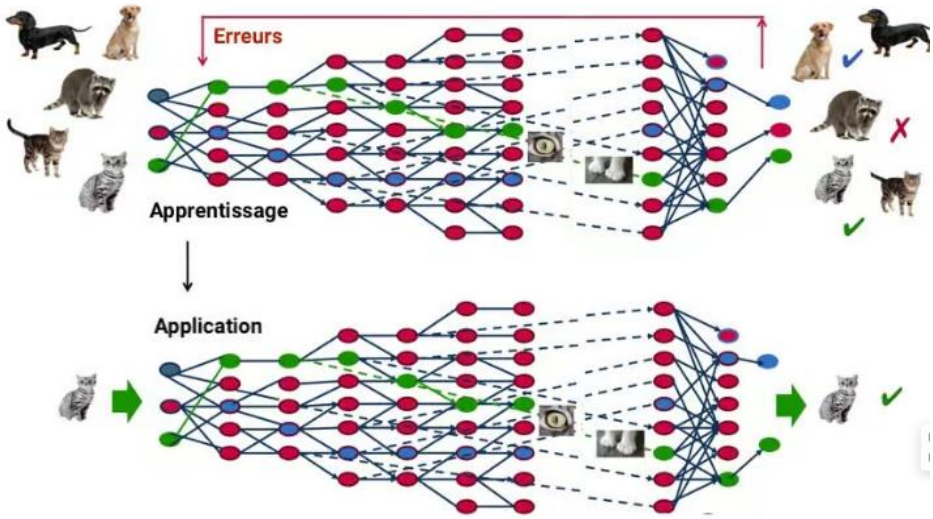

Figure 6 : Fonctionnement d'un modèle de deep learning[19]

1.1.5.4.6. Approche transfert Learning

Le Transfer Learning s'est largement développé grâce à l'avancée du Deep Learning. Les modèles de ce domaine demandent généralement beaucoup de puissance de calcul et des ressources considérables.

Cependant, en partant de modèles pré-entraînés, le Transfer Learning offre la possibilité de concevoir rapidement des solutions performantes pour traiter des problèmes complexes, notamment en vision par ordinateur ou en traitement automatique du langage. Il est particulièrement utile pour des tâches complexes comme la reconnaissance d'objets ou de piétons, en limitant le besoin de données spécifiques au nouveau domaine [15]. L'approche transfert Learning est illustrée dans la figure 7 ci-dessous qui montre la différence entre l'approche traditionnelle et celle du transfer Learning.

Dans l'approche traditionnelle, chaque système d'apprentissage est entraîné séparément sur ses propres données, sans tirer parti des connaissances acquises par d'autres systèmes. Cela signifie que le modèle commence de zéro pour chaque nouveau jeu de données, ce qui peut être coûteux en termes de temps et de ressources. Alors que l'approche par transfer Learning permet de réutiliser les connaissances acquises lors de l'entraînement sur un premier jeu de données pour faciliter et améliorer l'entraînement sur un second jeu de données.

[19] https://www.futura-sciences.com/tech/definitions/intelligence-artificielle-deep-learning-17262/

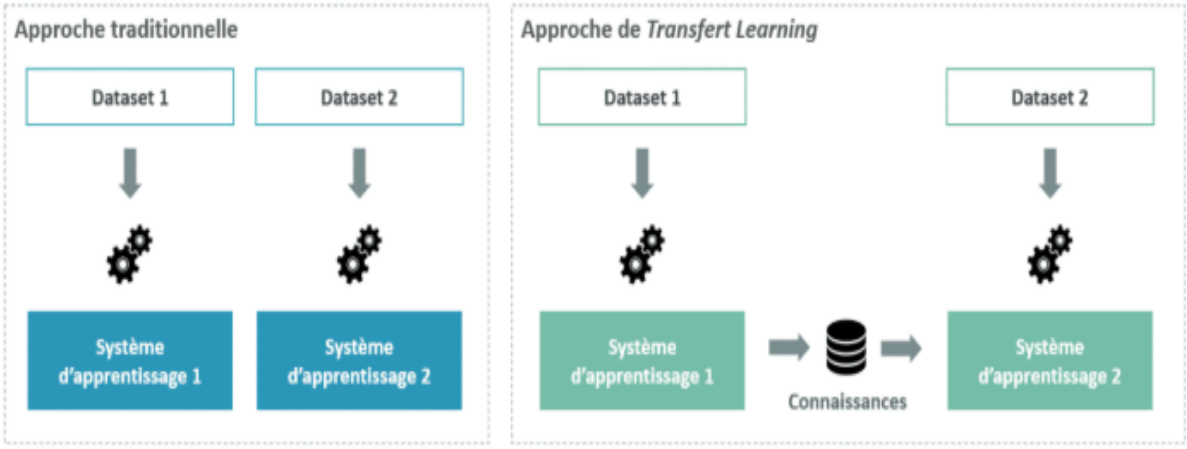

Figure 7 : Approche par Transfert learning[20]

Tableau 1 : Tableau de comparaison des différentes approche de machine Learning

ML	Description	Atouts	Limites
ML Supervisé	Méthode qui utilise des données étiquetées pour entraîner un modèle à prédire des résultats pour des données non étiquetées.	Très précis pour les tâches spécifiques (classification, régression). Adapté pour des problèmes bien définis avec beaucoup de données étiquetées.	Dépend fortement de la qualité et de la quantité des données étiquetées. Coût élevé pour l'étiquetage manuel.
ML non supervisé	Technique qui apprend des structures cachées ou des relations dans des données non étiquetées.	Utile pour explorer des données non structurées. Permet de découvrir des motifs et des regroupements non évidents.	Ne fournit pas de sortie directement interprétable. Moins précis sans étiquettes explicites.
ML semi supervisée	Combine un petit ensemble de données étiquetées avec un grand ensemble de données non étiquetées pour améliorer l'apprentissage.	Réduit le besoin de données étiquetées. Bonne performance dans les contextes où l'étiquetage est coûteux ou difficile.	Complexité accrue dans l'implémentation. Dépend de la qualité des données étiquetées disponibles.

[20] https://datascientest.com/transfer-learning

ML par renforcement	Modèle basé sur l'interaction entre un agent et un environnement, où l'agent apprend par essais-erreurs en recevant des récompenses ou des pénalités.	Très efficace pour des tâches séquentielles comme les jeux, la robotique ou la navigation. Permet d'apprendre des stratégies optimales dans des environnements dynamiques.	Longs temps d'entraînement. Nécessite une forte puissance de calcul et des environnements bien simulés.
ML par deep learning	Sous-domaine du machine learning utilisant des réseaux neuronaux profonds pour modéliser des problèmes complexes, comme la vision ou le traitement du langage naturel.	Excellente performance pour les données volumineuses et complexes. Capacité à extraire des caractéristiques pertinentes sans intervention humaine.	Fortement dépendant de la quantité et de la qualité des données. Nécessite des ressources informatiques élevées et peut manquer d'explicabilité.
ML par transfert learning	Technique qui utilise un modèle pré-entraîné sur un domaine source pour améliorer les performances sur un domaine cible, en ajustant les connaissances acquises.	Réduit le besoin de données dans le domaine cible. Accélère l'entraînement et améliore les performances dans des environnements similaires.	Moins efficace si le domaine cible est trop différent du domaine source. Limité dans les contextes spécifiques ou très différents.

- **Synthèse tableau**

Le Machine Learning supervisé est précis mais exige des données étiquetées, tandis que le non supervisé explore des motifs cachés sans étiquettes. Le semi-supervisé combine les deux pour réduire les besoins en étiquetage. L'apprentissage par renforcement apprend par interactions et récompenses, idéal pour les environnements dynamiques. Le deep Learning traite efficacement des données complexes grâce aux réseaux neuronaux, mais demande des ressources élevées. Enfin, le transfer Learning réutilise des modèles pré-entraînés pour réduire le besoin de données et accélérer l'apprentissage.

1.2. Problématique

La modération est un sujet passionnant[21] mais complexe. Le phénomène des commentaires toxiques met en évidence des failles significatives dans les mécanismes actuels de modération. Les mesures en place se révèlent souvent insuffisantes : elles permettent à des contenus nuisibles de persister ou, à l'inverse, suppriment certains messages sans tenir compte de leur contexte, renforçant l'idée d'une automatisation dénuée de nuances. La modération repose fréquemment sur des travailleurs précaires, chargés de signaler ou de vérifier manuellement les contenus, ce qui reflète une externalisation et une sous-évaluation de l'importance de cette tâche par les plateformes.

Dans ce contexte, intégrer le Machine Learning à la modération offre des perspectives intéressantes, mais le problème se situe sur la dépendance des algorithmes à s'entrainer sur des données claires et normalisées, ce qui devient particulièrement problématique dans des environnements linguistiques diversifiés comme celui du Sénégal. Les langues locales, comme le wolof ou le sérère, ne possèdent pas de normes écrites fixes, ce qui entraîne des variations importantes dans l'orthographe des mots. Cela complique leur traitement et peut affecter la précision des modèles d'analyse.

De plus, définir un seuil de toxicité adapté est un véritable défi. Alors que la modération humaine peut apporter un certain contexte, elle reste limitée par le volume de contenu à traiter et la subjectivité des interprétations. Les systèmes automatisés, quant à eux, peinent à interpréter les nuances culturelles ou contextuelles, aggravant ainsi les biais et les erreurs de classification. Ainsi, l'enjeu est de développer des approches hybrides qui combinent l'efficacité des outils technologiques et l'expertise humaine, tout en tenant compte des spécificités locales et culturelles.

[21] https://arobase.substack.com/p/les-defis-de-la-moderation-et-une

Chapitre 2 : Etat de l'art et positionnement

2.1. Etat de l'art

La détection de la toxicité dans les commentaires en ligne est un domaine de recherche actif, étant donné l'importance de lutter contre les abus sur les plateformes numériques. De nombreuses études ont été menées pour développer des modèles capables de détecter automatiquement les propos toxiques dans les discussions en ligne [16].

2.1.1. Modèles basés sur le Machine Learning supervisé

Les modèles d'apprentissage supervisé sont utilisés lorsque nous avons un ensemble de données étiquetées, par exemple chaque commentaire est accompagné d'une étiquette indiquant s'il est toxique ou non toxique. Ce modèle se base sur des exemples étiquetés pour apprendre à distinguer les caractéristiques d'un commentaire, afin de pouvoir prédire pour un nouveau commentaire s'il est toxique ou non [17]. Les principaux algorithmes d'apprentissage supervisé utilisés pour cette tâche :

- **Logistic Regression**[22]: La régression logistique est un modèle simple pour la classification binaire, pour distinguer les commentaires toxiques des commentaires non toxiques. Il fonctionne en estimant la probabilité qu'un commentaire appartienne à une classe donnée toxique ou non toxique en utilisant une fonction logistique pour transformer les scores en probabilités. Ce modèle prend en entrée des caractéristiques du texte, comme la fréquence des mots ou des n-grams pour faire la prédiction [18, 19, 28] ;
- **Support Vector Machine** [23]**(SVM)** : Les SVM sont des modèles qui cherchent à séparer les classes en trouvant un hyperplan optimal dans un espace de caractéristiques. Si nous avons des données non linéaires, un noyau peut être appliqué pour projeter les données dans un espace de plus grande dimension où une séparation linéaire devient possible. Les SVM sont adaptés aux situations où les classes ne sont pas linéairement séparables, ce qui les rend efficaces pour des tâches de classification complexes [19, 20, 21, 28] ;
- **Decision Tree** [24]**:** Les arbres de décision fonctionnent en divisant récursivement les données en sous-ensembles en fonction des caractéristiques les plus discriminantes, comme les mots ou n-grams présents dans un commentaire. À chaque niveau de l'arbre, une condition est appliquée pour séparer les données, jusqu'à ce que chaque feuille de

[22] https://aws.amazon.com/fr/what-is/logistic-regression/
[23] https://www.ibm.com/fr-fr/topics/support-vector-machine
[24] https://helios2.mi.parisdescartes.fr/~lomn/Cours/DM/Material/ComplementsCours/decisiontree.pdf

l'arbre contienne un commentaire classé comme toxique ou non toxique. Cette approche est intuitive et permet d'expliquer facilement les décisions prises par le modèle [20, 28] ;

- **Random Forest** [25]: Random Forest est un ensemble d'arbres de décision qui génèrent plusieurs arbres de manière aléatoire en échantillonnant les données et les caractéristiques. Les prédictions de ces arbres sont ensuite agrégées par un vote majoritaire pour classer un commentaire comme toxique ou non toxique. Cette méthode vise à réduire le risque de sur apprentissage observé dans les arbres de décision individuels. Le modèle Random Forest est robustes à la variance et capables de capturer des relations complexes entre les caractéristiques [20, 28] ;
- **Naive Bayes** [26]**:** Le modèle Naive Bayes reposent sur le théorème de Bayes et calcule la probabilité qu'un commentaire appartienne à une classe toxique en fonction des probabilités conditionnelles des caractéristiques du texte pour donner la classe. Ce modèle repose sur l'hypothèse naïve d'indépendance entre les caractéristiques, ainsi il simplifie les calculs tout en permettant une classification efficace [20].

[25] https://france.devoteam.com/paroles-dexperts/algorithme-n2-comprendre-comment-fonctionne-un-random-forest-en-5-min/#:~:text=Random%20forest%20signifie%20%C2%AB%20for%C3%AAt%20al%C3%A9atoire,il%20produit%20des%20r%C3%A9sultats%20g%C3%A9n%C3%A9ralisables.

[26] https://www.ibm.com/fr-fr/topics/naive-bayes#:~:text=Le%20classificateur%20Bayes%20na%C3%AFf%20est,effectuer%20ces%20t%C3%A2ches%20de%20classification.

–

Tableau 2 : Tableau récapitulatif des Algorithmes d'apprentissage supervisé pour la détection de commentaires toxique en ligne

Modèle	Algorithme	Avantages	Inconvénients
Apprentissage Supervisé	**Régression Logistique**	Simple, rapide, efficace pour les tâches binaires.	Moins performant pour des tâches complexes de classification de texte.
	SVM (Support Vector Machine)	Puissant pour des données non linéaires. Capable de gérer des espaces de caractéristiques complexes.	Sensible au choix du noyau et aux paramètres. Moins efficace avec des grandes quantités de données.
	Arbres de Décision	Facile à interpréter, intuitif, rapide à entraîner.	Susceptible au sur-apprentissage (overfitting), faible performance sur des données complexes.
	Random Forest	Robuste au sur-apprentissage, gestion des relations complexes.	Moins interprétable, nécessite plus de ressources en calcul.
	Naive Bayes	Simple, rapide, fonctionne bien avec de petites quantités de données.	Hypothèse d'indépendance souvent irréaliste, performance limitée sur des tâches complexes.

2.1.2. Modèles basés sur le Machine Learning non supervisé

Les modèles d'apprentissage non supervisé sont utilisés lorsque les données ne sont pas étiquetées pour entraîner le modèle. L'objectif est de permettre au modèle de découvrir des structures, des motifs ou des regroupements au sein des données sans supervision explicite. En ce qui concerne la modération des commentaires en ligne, les modèles d'apprentissage non supervisé sont précieux car ils permettent de détecter des structures, des comportements récurrents ou des anomalies sans nécessiter d'étiquettes de données préexistantes. Le clustering, l'apprentissage de représentation et la détection d'anomalies offrent des outils pour explorer et

comprendre les données textuelles de manière plus flexible en permettant de repérer de manière proactive les commentaires toxiques ou non [22, 25].

Les principaux algorithmes d'apprentissage non supervisé utilisés pour cette tâche :

- **K-means [27]:** est l'algorithme de clustering le plus utilisé. Il divise les données en K clusters en minimisant la variance intra-cluster. On pourrait l'utiliser pour regrouper des commentaires ayant des mots ou des expressions similaires pour ensuite identifier des groupes de commentaires toxiques ou non [6] ;
- **Word2Vec [28]**: Word2Vec est un modèle d'apprentissage non supervisé qui apprend des représentations vectorielles des mots à partir de grands corpus de textes. Ces vecteurs permettent de capter des relations sémantiques entre les mots. Par exemple, les mots comme "haine", "insulte" et "violence" auront des représentations similaires dans l'espace vectoriel. Cette méthode peut être utilisée pour calculer la similarité entre les commentaires et identifier des schémas de toxicité ou de langage agressif [23] ;
- **Isolation Forest[29]**: L'Isolation Forest est un algorithme d'apprentissage non supervisé conçu pour la détection d'anomalies. Il fonctionne en construisant plusieurs arbres qui "isolent" des points de données, et ceux qui sont difficiles à isoler sont considérés comme des anomalies [24].

Tableau 3 : Tableau récapitulatif des Algorithmes d'apprentissage non supervisé pour la détection de commentaires toxique en ligne

Modèle	**Algorithme**	**Avantages**	**Inconvénients**
Apprentissage Non Supervisé	**K-means**	Facile à implémenter, rapide, efficace sur des données linéaires.	Sensible à l'initialisation, nécessite de spécifier K, difficile avec des données non linéaires.
	Word2Vec	Capture bien les relations sémantiques entre les mots.	Nécessite de grandes quantités de données, pas directement une méthode de classification.
	Isolation Forest	Efficace pour la détection d'anomalies, ne nécessite pas d'étiquettes.	Moins précis pour la classification de texte standard.

[27] https://openclassrooms.com/fr/courses/4525281-realisez-une-analyse-exploratoire-de-donnees/5177935-decouvrez-l-algorithme-k-means
[28] https://datascientest.com/nlp-word-embedding-word2vec
[29] https://scikit-learn.org/1.5/modules/generated/sklearn.ensemble.IsolationForest.html

2.1.3. Modèles basés sur le Machine Learning semi supervisé

Les modèles semi-supervisés combinent les avantages de l'apprentissage supervisé et non supervisé. Ces modèles sont particulièrement utiles lorsque l'on dispose d'une petite quantité de données étiquetées et d'une grande quantité de données non étiquetées. Pour la modération des commentaires en ligne, cela représente un scénario fréquent, car l'étiquetage manuel de chaque commentaire peut être coûteux et laborieux. L'idée principale des modèles semi-supervisés est d'utiliser les données non étiquetées pour améliorer les performances du modèle tout en s'appuyant sur les données étiquetées pour guider l'apprentissage.

Cette approche permet d'augmenter l'efficacité des modèles tout en réduisant la dépendance aux données étiquetées [25]. Les principaux algorithmes d'apprentissage semi-supervisés utilisés pour cette tâche :

- **Label Propagation**[30]: Le modèle de propagation de labels est une approche semi-supervisée où les labels des exemples étiquetés sont "propagés" à travers un graphe construit à partir des données. Chaque nœud du graphe représente un exemple et des arêtes sont établies entre les exemples similaires c'est-à-dire les commentaires qui sont sémantiquement proches. Les étiquettes des nœuds étiquetés sont propagées à leurs voisins dans le graphe, au final des étiquettes sont assignées à tous les nœuds, y compris les données non étiquetées [26, 27] ;
- **Co-training**[31]**:** Le Co-training est une technique semi-supervisée où deux modèles sont formés simultanément, chacun sur un ensemble différent de caractéristiques extraites des données. Ces modèles peuvent ensuite s'échanger des labels sur les exemples non étiquetés qu'ils classifient avec une grande confiance. L'idée est même si les données non étiquetées ne sont pas directement disponibles, les deux modèles peuvent s'entraider en étiquetant les données qu'ils considèrent comme les plus fiables [28].

[30] https://scikit-learn.org/dev/modules/generated/sklearn.semi_supervised.LabelPropagation.html
[31] https://medium.com/@data-overload/co-training-unveiled-harnessing-multifaceted-learning-for-semisupervised-success-3952de32e22d

Tableau 4 : Tableau récapitulatif des Algorithmes d'apprentissage semi-supervisé pour la détection de commentaires toxique en ligne

Modèle	Algorithme	Avantages	Inconvénients
Apprentissage Semi-Supervisé	**Label Propagation**	Moins de données étiquetées nécessaires, exploite les relations entre données non étiquetées.	Sensible à la qualité du graphe, nécessite une structure de données bien définie.
	Co-training	Utilise efficacement les données non étiquetées, améliore les performances avec peu de données étiquetées.	Complexe à mettre en œuvre, nécessite deux ensembles de caractéristiques.

2.1.4. Modèles basés sur le deep learning

Les modèles basés sur le Deep Learning utilisent des réseaux de neurones à plusieurs couches pour apprendre des représentations hiérarchiques des données et capturer des relations complexes. Ces modèles sont efficaces pour traiter des données non structurées, comme le texte, l'image ou la voix. Les modèles de Deep Learning sont capables de comprendre des contextes complexes et d'extraire des caractéristiques sémantiques avancées des textes, ce qui les rend idéaux pour détecter du contenu toxique, haineux ou nuisible. Ces algorithmes apprennent à partir de grandes quantités de données d'entraînement pour classifier les commentaires et améliorer la gestion du contenu en ligne [29]. Les principaux algorithmes d'apprentissage par Deep Learning utilisés pour cette tâche :

- **Convolutional Neural Network** [32] : Les CNN sont utilisés pour extraire des motifs locaux dans des données structurées comme les séquences de texte. Les filtres appliqués par un CNN peuvent détecter des motifs linguistiques significatifs par des combinaisons de n-grams qui indiquent des commentaires toxiques ou non [30] ;
- **Recurrent Neural Network** [33] : Les RNN sont bien adaptés pour les tâches de traitement de séquences de texte, car ils prennent en compte l'ordre des mots dans une phrase. Cependant, les RNN classiques peuvent avoir des difficultés avec des séquences

[32] https://www.ibm.com/fr-fr/topics/convolutional-neural-networks
[33] https://www.ibm.com/fr-fr/topics/recurrent-neural-networks

longues, ce qui a conduit au développement de variantes plus avancées comme les LSTM ou les GRU [23] ;

- **LSTM[34] (Long Short-Term Memory)** : Les LSTM sont une variante des RNN, conçue pour résoudre le problème de vanishing gradient et pour apprendre des dépendances à long terme dans des séquences de texte [23] ;
- **Transformers**[35] : Les Transformers ont révolutionné le traitement du langage naturel. Contrairement aux RNN et LSTM, les Transformers utilisent un mécanisme d'attention pour capturer les relations entre tous les mots d'une séquence simultanément, ce qui est particulièrement efficace pour des tâches de compréhension de texte complexes, comme la détection de la toxicité dans un commentaire [30].

Tableau 5 : Tableau récapitulatif du deep Learning pour la détection de commentaires toxique en ligne

Modèle	**Algorithme**	**Avantages**	**Inconvénients**
Deep Learning	**CNN**	Bonne performance pour les séquences de texte structurées, capture bien les motifs sémantiques.	Moins efficace pour des textes très longs ou avec des contextes complexes.
	RNN	Prend en compte l'ordre des mots, adapté aux séquences de texte.	Difficulté avec les longues séquences, problèmes de vanishing gradient.
	LSTM	Gère les dépendances longues et complexes.	Plus complexe à entraîner, nécessite beaucoup de données et de ressources.
	Transformers	Excellente performance sur des tâches de compréhension du texte, capture bien les relations contextuelles.	Très coûteux en calcul, nécessite des ressources importantes.

[34] https://datascientest.com/long-short-term-memory-tout-savoir#:~:text=Les%20LSTM%20introduisent%20une%20nouvelle,informations%20sur%20une%20p%C3%A9riode%20%C3%A9tendue.&text=La%20cellule%20m%C3%A9moire%20d'un,et%20une%20porte%20d'oubli.

[35] https://blent.ai/blog/a/transformers-deep-learning

2.1.5. Modèles basés sur le transfer learning

Le transfert Learning[36] consiste à réutiliser un modèle préalablement entraîné sur de grandes quantités de données et à l'adapter à des tâches spécifiques, comme la modération des commentaires en ligne. Ce processus permet de tirer parti de modèles généraux pré-entraînés sur des tâches de traitement du langage naturel et de les affiner pour la détection de la toxicité dans les commentaires. Dans ce contexte, les modèles basés sur les Transformers sont couramment utilisés, car ils peuvent être adaptés pour classer efficacement les commentaires en ligne, même dans des scénarios avec des données déséquilibrées [31]. Les principaux algorithmes d'apprentissage par transfert Learning utilisés pour cette tâche :

- **BERT[37] (Bidirectional Encoder Representations from Transformers)** : BERT est l'un des modèles les plus populaires dans le traitement du langage naturel et est particulièrement efficace dans les tâches de classification de texte [30] ;
- **RoBERTa[38] (A Robustly Optimized BERT Pretraining Approach)** RoBERTa est une variante de BERT qui a été améliorée par des optimisations dans le processus de pré-entraînement [30] ;
- **DistilBERT[39] :** DistilBERT est une version plus légère et plus rapide de BERT, conçue pour être plus efficace tout en maintenant une grande partie des performances de BERT [30] ;
- **ALBERT[40] (A Lite BERT) :** ALBERT est une version allégée de BERT, qui réduit le nombre de paramètres tout en maintenant des performances comparables [30] ;
- **T5[41] (Text-to-Text Transfer Transformer) :** T5 est un modèle Transformer qui reformule toutes les tâches NLP comme un problème de transformation de texte en texte. Cela permet une plus grande flexibilité pour de nombreuses applications [32].

[36] https://www.cyber-management-school.com/ecole/les-fondamentaux-de-la-cybersecurite/transfer-learning-definition/#:~:text=Le%20Transfer%20Learning%2C%20ou%20apprentissage,%C3%A0%20accomplir%20un%20sc%C3%A9nario%20diff%C3%A9rent.
[37] https://datascientest.com/bert-un-outil-de-traitement-du-langage-innovant
[38] https://www.intelligence-artificielle-school.com/ecole/technologies/abert-modele-nlp/
[39] https://www.sciencedirect.com/topics/computer-science/distilbert
[40] https://paperswithcode.com/method/albert
[41] https://formation-redaction-web.com/algorithme-google-mum/#:~:text=Le%20nouvel%20algorithme%20de%20Google,information%20par%20le%20cerveau%20humain.

Tableau 6 : Tableau récapitulatif du Transfert Learning pour la détection de commentaires toxique en ligne

Modèle	Algorithme	Avantages	Inconvénients
Transfert Learning	**BERT**	Excellente capacité de compréhension du contexte, très précis pour la classification de texte.	Coût de calcul élevé, nécessite une fine-tuning [42] spécifique.
	RoBERT	Plus robuste que BERT pour les tâches complexes.	Nécessite plus de données et de ressources pour le fine-tuning.
	DistilBERT	Moins cher en calcul, conserve la plupart des performances de BERT.	Moins performant sur des tâches très complexes.
	ALBERT	Moins de mémoire nécessaire, bonne performance avec moins de ressources.	Moins performant que BERT pour des tâches très complexes.
	T5	Grande flexibilité, excellente performance sur diverses tâches NLP.	Complexité élevée, nécessite un fine-tuning spécifique pour chaque tâche.

2.2. Positionnement

Face à notre problématique, nous avons opté pour une approche mixte combinant Deep Learning et Transfert Learning. Cette approche permet de tirer parti des avantages des deux méthodes pour mieux gérer la diversité et la subtilité des commentaires.

Nous avons choisi les Réseaux de Neurones Récurrents (RNN)**,** notamment la variante LSTM, comme algorithme de Deep Learning en raison de sa capacité à traiter les dépendances contextuelles dans des séquences de texte. Ce modèle est efficace pour analyser l'ordre des mots et saisir les relations entre eux, ce qui est essentiel pour détecter la toxicité dans des commentaires dont le sens dépend fortement du contexte.

Pour le Transfer Learning**,** nous avons opté pour BERT (Bidirectional Encoder Representations from Transformers), un modèle de pointe qui permet une compréhension

[42] https://datascientest.com/fine-tuning-tout-savoir

bidirectionnelle du contexte. Cette approche est particulièrement utile pour saisir les subtilités du langage, comme le sarcasme ou les expressions implicites, qui sont souvent des indicateurs de toxicité dans les commentaires en ligne. En combinant ces deux approches, nous visons à maximiser la compréhension contextuelle et la détection fine des nuances dans les commentaires, pour rendre notre modèle plus robuste et performant pour la modération de contenu en ligne.

Chapitre 3 : Extraction des commentaires

3.1. Acquisition des données de tests par web scraping

3.1.1. Nos sources de données

Les données utilisées pour la mise en œuvre de notre projet de mémoire proviennent pour la plupart de la plateforme YouTube. Pour ce faire, nous avons conçu des algorithmes permettant d'extraire les commentaires publiés sur ces plateformes. Afin de recueillir des données représentatives, nous avons opté pour deux approches pour faire le scraping des commentaires.

La première approche consiste à extraire les commentaires publiés sur la chaine à la fin de la diffusion en direct. La deuxième approche, quant à elle, permet d'extraire les commentaires en temps réel publiés par les spectateurs durant la diffusion en direct de l'émission.

Ces deux approches nous ont permis de constituer un dataset comprenant à la fois des commentaires de spectateurs post-émission et des réactions en direct. Ce dataset nous servira de base pour l'entraînement de notre modèle de modération automatique des commentaires toxiques sur les réseaux sociaux.

3.1.2. Outils et technologies nécessaires

Pour bien réaliser notre projet, nous avons utilisé une série d'outils et de technologie qui nous ont permis de collecter, analyser et traiter efficacement les données. On peut citer :

- **Python[43] :** Le langage de programmation Python a été choisi pour sa simplicité et sa puissance dans le domaine de notre projet de mémoire. Il est largement utilisé dans le domaine du traitement du langage naturel, ce qui est un choix adapté pour notre projet de modération des commentaires. Python dispose également de nombreuses bibliothèques comme BeautifulSoup, Sélénium, Requests …qui facilite l'extraction de données depuis des plateformes web. Les bibliothèques comme Pandas et Numpy sont utilisées pour la gestion des données par contre scikit-learn et TensorFlow, Torch sont utilisés pour l'entraînement et la mise en œuvre des modèles de modération ;
- **Anaconda[44] :** Anaconda est un environnement de développement intégré (IDE) populaire pour la gestion des projets Python. Il permet de gérer facilement les environnements virtuels et les dépendances nécessaires au projet, cela nous permet d'obtenir un environnement de travail isolé et garantir la compatibilité des différentes bibliothèques utilisées dans le projet. On y trouve également dans anaconda des outils

[43] https://blog.hubspot.fr/website/python#:~:text=Python%20est%20un%20langage%20de%20programmation%20open%2Dsource%20tr%C3%A8s%20populaire,continu%20par%20la%20communaut%C3%A9%20Python.

[44] https://datascientest.com/installer-anaconda-tout-savoir

comme Jupyter Notebook utile pour l'analyse interactive des données et le test de nos algorithmes de traitement des commentaires ;

- **Visual Studio Code [45]:** VSCode est un éditeur de code principal que nous avons utilisé pour faire du code de notre projet. Il offre de nombreuses extensions et de fonctionnalités qui améliore la coloration syntaxique, le débogage intégré, la gestion de version via Git. Grâce à sa flexibilité et son intégration avec ses extensions Python, VsCode facilite le développement des scripts de scraping le traitement des données et l'intégration des différents modèles d'apprentissage automatique dans la modération.
- **Google Colab :** Google Colab est une plateforme cloud gratuite qui nous permet de développer et d'exécuter du code directement dans un environnement en ligne. Elle offre des ressources de calcul puissant, notamment des GPU qui nous ont permis d'entraîner et de tester rapidement notre modèle de modération des commentaires toxiques. De plus l'accès facile aux fichiers et à Google Drive permet collaboration fluide et un stockage sécurisé des données et des résultats ;
- **Google Cloud Console**[46] **:** Google Cloud Console est la plateforme intermédiaire utilisée pour l'acquisition de clé API You Tube pour l'extraction des commentaires.
- **Google sheets :** Google sheets est un outil de feuille de calcul accessible en ligne il nous sert d'outil d'annotation pour l'ensemble des données collectées. Son avantage est la possibilité de collaboration avec les volontaires, qui nous aident à discuter sur l'état de label d'un commentaire ;
- **Google drive :** est un service de stockage cloud qui permet de sauvegarder, partager et accéder à des fichiers depuis n'importe quel appareil. Il offre une intégration fluide avec d'autres outils Google comme Docs, Sheets et Slides, facilitant la collaboration en temps réel. Avec une capacité de stockage gratuite de 15 Go (extensible via abonnement), il est idéal pour stocker des données. Son accessibilité et ses fonctionnalités de partage en font un outil essentiel pour le travail collaboratif.

Ces outils et technologies, nous ont permis de concevoir un système capable d'extraire, traiter, analyser les commentaires en lignes afin de les modérer les commentaires sur les réseaux sociaux.

Python a été au cœur de notre projet avec ses bibliothèques dédiées à l'extraction et l'analyse de données tandis qu'Anaconda, VsCode et Google Colab ont fourni les environnements

[45] https://code.visualstudio.com/
[46] https://cloud.google.com/storage/docs/cloud-console?hl=fr

nécessaires au développement et à l'entraînement de notre modèle pour la modération des commentaires toxiques.

3.1.3. Mise en œuvre de la collecte

Dans ce mémoire, pour collecter les données nécessaires à l'analyse de la toxicité des commentaires en ligne, nous avons utilisé une approche automatisée en exploitant l'API YouTube Data API v3.

3.1.3.1. Choix de la plateforme YouTube

Dans le cadre de cette étude, YouTube a été retenu comme source principale d'extraction de données en raison de sa popularité mondiale et de sa capacité à fournir des interactions textuelles riches, notamment via les espaces commentaires associés aux vidéos et diffusions en direct.

En tant que première plateforme de visionnage de vidéos au niveau mondial[47], YouTube offre une diversité d'opinions et de points de vue exprimés par des utilisateurs provenant de différents contextes socioculturels. De plus, au Sénégal, YouTube est classée comme le deuxième site le plus visité, avec plus de 40,71 millions de visites mensuelles[48]. Ces caractéristiques font de lui un choix pertinent pour collecter des données à grande échelle tout en conservant leur authenticité.

3.1.3.2. Ciblage des vidéos pour la collecte de commentaires

Une attention particulière a été portée au ciblage des vidéos à analyser. Cette étape s'est avérée cruciale, car elle conditionne la qualité et la représentativité des données collectées. Pour traiter efficacement la problématique, nous avons jugé pertinent de recueillir à la fois des commentaires positifs et négatifs, de manière à capturer une variété de discours représentatifs des interactions en ligne.

Après des expérimentations approfondies, il est apparu que la majorité des vidéos populaires sur des plateformes telles que YouTube sont associées à des commentaires globalement positifs. Ce phénomène est particulièrement observé pour les séries télévisées, les films, et les clips musicaux, qui constituent les contenus les plus visionnés et commentés. Dans ces catégories, les réactions sont généralement unanimes et majoritairement émotionnelles, reflétant un consensus des spectateurs autour du contenu proposé.

En revanche, les commentaires toxiques sont significativement plus fréquents dans un autre type de contenu : les émissions abordant des sujets d'actualité politique ou sociale. Par exemple,

[47] https://riverside.fm/blog/video-platforms
[48] : https://fr.semrush.com/trending-websites/sn/online-services

des programmes tels que *Jakarlo* (diffusé sur TFM), *Ndoumbelane* (diffusé sur SEN TV) … recueillent une proportion notable de commentaires polarisés, parfois marqués par la colère ou l'agressivité. Un cas particulièrement révélateur est la réaction des internautes à la sortie médiatique de Cheikhou Omar Diagne sur le thème des tirailleurs sénégalais. Cette intervention a suscité une vive polémique, provoquant une vague de colère parmi les Sénégalais, sensibles à ce sujet historique et identitaire.

Ces observations mettent en évidence que les sujets sensibles, qu'ils soient d'ordre politique ou social, génèrent des tensions et exacerbent les émotions dans les espaces de discussion en ligne. Ces vidéos constituent donc des sources privilégiées pour collecter des commentaires toxiques, essentiels dans notre projet.

En somme, ce ciblage rigoureux des vidéos nous a permis de collecter des données diversifiées et équilibrées, offrant ainsi une base solide pour l'analyse et la modélisation des commentaires en ligne.

3.1.3.3. L'utilisation de l'API YouTube Data

Pour automatiser la collecte des données, nous avons besoin d'une clé API : YouTube Data API v3. Cet outil offre un accès programmatique à de nombreuses fonctionnalités de la plateforme, notamment la récupération des commentaires publiés sous les vidéos ou durant les diffusions en direct. Cette API est bien documentée et largement utilisée dans des recherches impliquant des analyses de contenus textuels ou multimédias. L'utilisation de l'API nécessite une clé d'authentification unique, que l'on a obtenue via le tableau de bord de la plateforme Google Cloud. Cette clé est acquise suite à l'authentification sur garantit que l'accès aux données respecte les règles de confidentialité et les quotas imposés par YouTube.

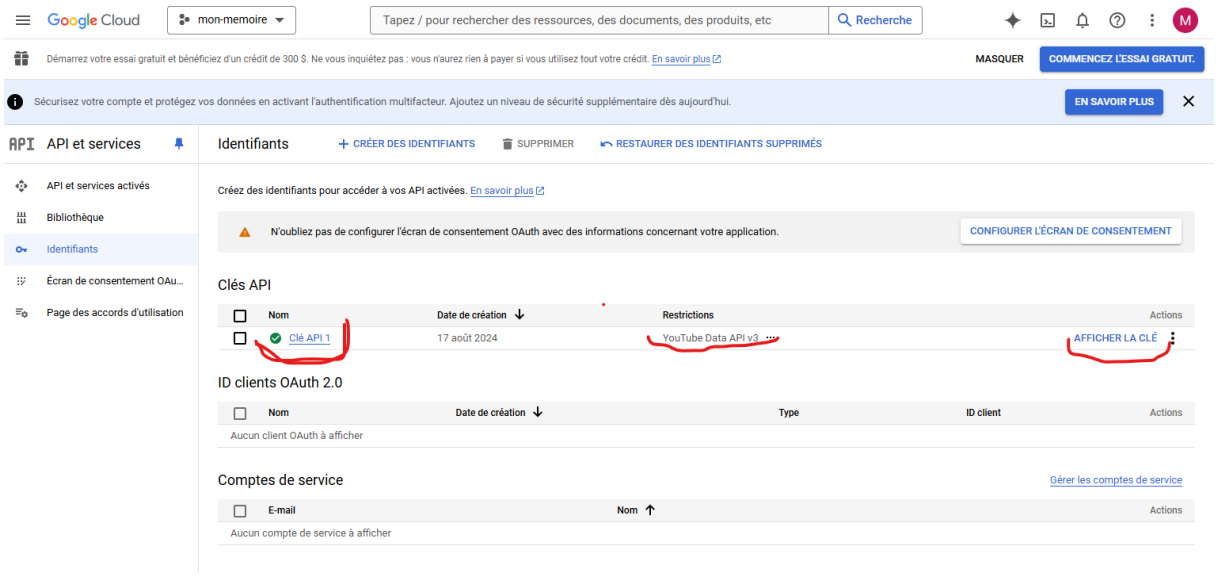

Figure 8 : Capture d'écran de l'acquisition de la clé API

3.1.3.4. Collecte des commentaires des vidéos enregistrées

Les vidéos enregistrées constituent une source importante pour capturer des interactions réfléchies et permanentes. La méthodologie adoptée pour leur collecte est détaillée ci-dessous :

- **Identification des vidéos pertinentes :** Les vidéos ont été sélectionnées en fonction de leur popularité et de leur pertinence suite à un visionnage des commentaires qui sont en lien avec les thématiques abordées dans cette recherche ;
- **Développement du script Python :** Le script utilise la bibliothèque google-api-python-client pour interagir avec l'API YouTube Data V3. Une fois la connexion établie à l'aide de la clé API, les commentaires des vidéos sont extraits par lots et sauvegardés localement ;
- **Sauvegarde et structuration des données :** Les commentaires sont stockés dans un fichier CSV, structuré avec des colonnes correspondant aux métadonnées des vidéos et au contenu textuel des commentaires.

3.1.3.5. Collecte des commentaires des vidéos en direct

Les diffusions en direct offrent une perspective unique grâce à leur caractère spontané et interactif. La collecte de ces données repose sur une méthode similaire, avec quelques adaptations pour gérer le flux en temps réel :

- **Identification des vidéos pertinentes en direct :** Pour ce qui est des vidéos transmises en direct, il existe un identifiant associé à une diffusion active unique. Ce dernier est visible sur l'url, juste après les caractères ''v= '' (voir figure 8). Cet identifiant est essentiel pour collecter les commentaires échangés pendant le live ;

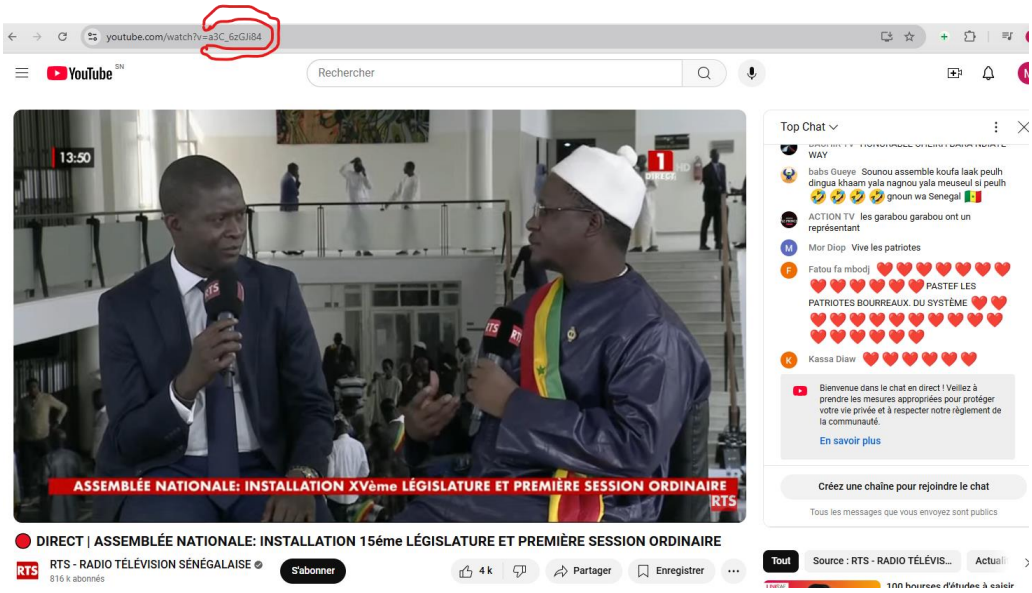

Figure 9 : Capture d'écran de l'id des vidéos en direct

- **Extraction et sauvegarde des messages :** Les messages publiés en temps réel sont récupérés à l'aide de requêtes itératives envoyées à l'API. Ces données sont sauvegardées en continu dans un fichier CSV pour garantir leur disponibilité ;
- **Automatisation et gestion des quotas :** Un délai est intégré entre chaque requête pour respecter les quotas imposés par l'API, tout en garantissant une collecte exhaustive.

3.1.3.6. Présentation des données

Les données collectées pour cette étude proviennent des commentaires extraits de la plateforme YouTube, à la fois sous des vidéos enregistrées et lors de diffusions en direct. Cette section présente les caractéristiques principales des données, notamment leur taille, leur structure, et leur type.

Description des données

Les données collectées ont été sauvegardées sous un format tabulaire dans un fichier CSV, facilitant leur exploitation ultérieure pour l'analyse. Les principales caractéristiques de ce fichier sont les suivantes :

- **Taille des données :** Le fichier contient un total de **3515 lignes** et **2 colonnes**. Chaque ligne représente un commentaire unique.
- **Colonnes principales :** Les colonnes du fichier CSV sont les suivantes :
  1- **Author :** Identifiant unique de la vidéo d'où provient le commentaire ;
  2- **Message :** Texte du commentaire publié par un utilisateur.
- **Type de données :** Les données sont principalement textuelles (commentaires) avec des métadonnées associées sous forme de chaînes de caractères ou d'horodatages.

La figure ci-dessous illustre un extrait du fichier CSV contenant les données collectées :

	Author	Message
0	mary ciss	:sanitizer::sanitizer::sanitizer::sanitizer::s...
1	Mouhammed_d	Eric fama😂 😂 😂
2	Morota dieng	2stv seul ✌️
3	Ahmadou Bamba Thioune	ama balde
4	GALASH 683	❤️ ❤️ ❤️ ❤️ ❤️ ❤️ ❤️ ❤️
...	...	...

Figure 10 : échantillon de commentaires extraits

La collecte a permis de capturer des interactions riches et variées. Les données incluent des commentaires de longueur variable, allant de simples réponses de quelques mots à des réflexions plus complexes et détaillées. Cette diversité garantit une représentativité dans les analyses qui seront réalisées.

En conclusion, le fichier CSV contient un corpus structuré qui représente fidèlement les échanges des utilisateurs sur YouTube, constituant une base solide pour les analyses qualitatives ultérieures.

3.2. Prétraitement des commentaires brutes

Le prétraitement est une étape cruciale dans la préparation des données pour une analyse efficace et une future labélisation. Voici les étapes principales suivies pour une bonne structuration des commentaires :

1. **Importation des bibliothèques nécessaires**

- **unicodedata** : Permet de manipuler les caractères Unicode, comme la gestion des accents.
- **re** : Permet d'utiliser des expressions régulières pour rechercher et manipuler des chaînes de caractères.

2. **Création d'une fonction clean_comment**

Cette fonction est conçue pour nettoyer les commentaires en plusieurs étapes :

- **Conversion en minuscules** : Les commentaires sont transformés en lettres minuscules pour uniformiser les données. Par exemple, *"AS SALAM MOU ALEY KOUM"* devient *"as salam mou aley koum"*.
- **Suppression des accents** : Les lettres accentuées sont remplacées par leur équivalent non accentué. Par exemple, *"é"* devient *"e"*. Cela garantit une cohérence dans les données, notamment pour l'analyse textuelle.
- **Suppression des emojis** : Tous les caractères Unicode représentant des symboles ou des pictogrammes sont supprimés. Cela évite les interférences avec l'analyse textuelle.
- **Suppression des caractères spéciaux** : Tous les caractères non alphanumériques (! `@`, `#`, etc.) sont retirés, sauf les espaces et les lettres. Cela simplifie les données en se concentrant uniquement sur le texte.
- **Suppression des espaces inutiles** : Les espaces superflus en début ou en fin de commentaire sont éliminés.

3. **Application de la fonction aux données**

Chaque commentaire de la colonne **"Message"** de la base de données est nettoyé en appliquant la fonction clean_comment. Cela remplace le contenu brut par une version plus épurée.

4. **Ajout d'une colonne label**

Une nouvelle colonne intitulée **"label"** est ajoutée aux données, initialisée à la valeur **1** pour tous les commentaires. Cette colonne est initiée pour faciliter la labélisation.

5. **Enregistrement des données nettoyées**

La base de données modifiée, contenant les commentaires nettoyés et la nouvelle colonne, est enregistrée dans un fichier CSV. Cela facilite le stockage et le partage des données pour les étapes suivantes du projet.

	Video ID	Commentaire	label
0	bKIT07SC7IM	machallah sonko	1
1	bKIT07SC7IM	octobre 2024 toujours la	1
2	bKIT07SC7IM	ndeysann xoolen ahmed ndoye \nmo metamorphosew...	1
3	bKIT07SC7IM	sacre dame mbodji	1
4	bKIT07SC7IM	eminent dame futur honorable tres pertinent	1
...	...	...	...
3345	3Z1cz5BTRUE	katal sa leufeul tounou ndeye feule domou raam	1
3346	3Z1cz5BTRUE	dagay doul	1
3347	3Z1cz5BTRUE	NaN	1
3348	3Z1cz5BTRUE	katal dayam	1
3349	3Z1cz5BTRUE	NaN	1

3350 rows × 3 columns

Figure 11 : échantillon de commentaires prétraités

3.3. Acquisition des données d'entrainement

3.3.1. Etiquetage de commentaires toxiques

L'étiquetage des données constitue une étape fondamentale pour la création d'un modèle efficace de détection des commentaires toxiques. Dans notre démarche, nous avons opté pour une méthode collaborative et minutieuse, facilitée par l'utilisation de Google Sheets.

Les commentaires collectés ont d'abord été importés dans une feuille Google Sheets. Cet outil a été choisi pour son accessibilité en ligne et ses fonctionnalités de partage en temps réel, permettant une collaboration fluide entre les membres de l'équipe. Chaque commentaire est initialement annoté par défaut avec une étiquette **1**, signifiant qu'il est considéré comme toxique.

Cependant, si un commentaire est jugé non toxique, son étiquette est modifiée pour prendre la valeur **0**, ce processus d'étiquetage est effectué ligne par ligne, avec une attention particulière au contexte des commentaires. Nous avons sollicité la collaboration de plusieurs personnes engagées et sensibilisées aux spécificités de cette tâche. Leur contribution a été essentielle pour interpréter correctement le contenu, en tenant compte des nuances culturelles, linguistiques et sociales propres à chaque commentaire.

Cette approche rigoureuse et collaborative nous a permis de garantir une annotation précise et fiable des données, offrant ainsi une base solide pour l'entraînement et l'évaluation de notre modèle de classification.

3.3.2. Autres Datasets étiquetés

Dans le cadre de l'amélioration et de l'enrichissement de notre base de données initiale pour la modération des commentaires toxiques, nous avons intégré plusieurs jeux de données complémentaires provenant de contributions externes et internes. Ces ajouts apportent une diversité notable et renforcent la robustesse des algorithmes que nous avons développés.

Une contribution significative a été réalisée par **Mame Sofietou GUEYE**, étudiante en Licence, qui a travaillé sur la « Mise en place d'une base de reconnaissance de commentaires toxiques pour une modération automatique » pour un total de **204 commentaires toxiques en ce moment**. Ces données sont particulièrement intéressantes en raison de leur classification en plusieurs catégories :

- **Insultes** ;
- **Propos violents** ;
- **Misogynie** ;
- **Racisme** ;
- **Diffamation** ;
- **Body shaming**.

Bien que cette catégorisation thématique soit précieuse pour des analyses spécifiques ou une modélisation future ciblée, notre travail s'est principalement focalisé sur la toxicité générale des commentaires. Cela signifie que nous avons considéré ces données dans leur globalité, sans

inclure les catégories dans l'entraînement de nos algorithmes actuels. Néanmoins, la richesse de cette collecte offre des opportunités d'analyse pour des travaux ultérieurs.

Par ailleurs, nous avons intégré des bases de données fournies par notre encadreur, le **Dr Edouard Ngor Sarr**. Ces données se composent de deux ensembles principaux :

- **Un corpus de 52 commentaires toxiques** : Ces données ont été directement collectées dans des contextes variés, et bien qu'elles soient relativement limitées en volume, elles complètent notre base existante en apportant une diversité contextuelle ;
- **Un ensemble structuré de mots issus de langues locales** : Ce dataset est particulièrement unique puisqu'il inclut des termes ou expressions souvent utilisés dans des langues locales, comme le wolof, le sérère ou le peulh. Ces mots, bien que différents dans leur structure des commentaires typiques des réseaux sociaux, permettent de capturer la toxicité dans un contexte linguistique sous-représenté.

Ces deux ensembles ont nécessité des ajustements pour être compatibles avec notre modèle. Notamment, le corpus de mots locaux a été transformé pour respecter le format des données initiales et a suivi le même processus de prétraitement. Cela inclut la normalisation des textes, la suppression des doublons et le traitement des caractères spécifiques.

L'intégration de ces datasets supplémentaires a renforcé de manière significative la diversité et la couverture de notre base de données. En particulier, ils nous ont permis :

- D'élargir le champ d'application de nos algorithmes, en incluant des formes de toxicité et des expressions propres à des contextes culturels variés ;
- De réduire les biais potentiels liés à la surreprésentation de certains types de données ou de langues dans la base initiale ;
- D'augmenter la capacité de nos modèles à détecter des formes subtiles ou contextuelles de toxicité.

Enfin, ces apports se sont révélés essentiels pour améliorer la précision de nos algorithmes. En combinant ces données à notre dataset principal, nous avons constitué une base de données solide, capable de répondre aux exigences d'un système de modération performant dans des environnements multilingues et multiculturels, en termes de diversité linguistique et culturelle, renforçant ainsi la robustesse et la précision de nos algorithmes. Elles représentent un levier important pour affiner la détection de la toxicité, notamment dans des contextes variés ou dans des langues sous-représentées dans les corpus classiques. En combinant ces différentes sources, nous disposons d'un ensemble de données étoffé et adapté aux spécificités de notre projet.

Chapitre 4 : Classification des commentaires toxiques

4.1. Prétraitement des données

Avant d'utiliser les données collectées pour l'entraînement de notre modèle de détection de commentaires toxiques, un processus rigoureux de prétraitement a été effectué. L'objectif principal était de garantir la qualité des données et de maximiser la pertinence des informations extraites pour améliorer la performance du modèle.

4.1.1. Validation de l'intégrité des données

La première étape du prétraitement a consisté à s'assurer qu'il n'existait aucune donnée manquante dans notre dataset. Toutes les observations ont été soigneusement vérifiées, et les lignes ou colonnes contenant des valeurs manquantes ont été traitées selon des méthodes standards, telles que la suppression ou l'imputation, pour maintenir la cohérence des données.

4.1.2. Exploration des mots les plus fréquents

Une analyse exploratoire approfondie a été réalisée pour examiner la distribution des mots dans l'ensemble des données. Cette fouille de données avait pour but de comprendre les termes les plus souvent captés dans les commentaires et d'évaluer leur pertinence pour la détection de la toxicité. Pour cela, nous avons séparé notre dataset en deux classes distinctes à savoir **Commentaires toxiques et Commentaires non toxiques.**

Pour chaque classe, une analyse des mots les plus fréquents a été effectuée afin de détecter des motifs récurrents et de valider leur pertinence en lien avec la toxicité.

4.1.3. Découverte et nettoyage des noms de personnes

Au cours de cette analyse, nous avons constaté que certains mots revenaient fréquemment dans les deux classes de labels, sans distinction notable. Ces mots incluent, entre autres :

- **'tirailleurs'**,
- **'sonko'**,
- **'bougane'**,
- **'moussa'**,
- **'birima'**,
- **'birma'**,
- **'maître'**,
- **'diouf'**.

Figure 12 : Top 20 des mots les plus fréquents avant prétraitement

Une observation attentive a révélé que ces termes correspondent majoritairement à des **noms de personnes** ou à des **noms communs associés à des individus**. Bien que ces mots puissent être importants dans un contexte général, leur récurrence dans les deux classes les rend non pertinents pour la détection de la toxicité des commentaires. En effet, ces mots ne permettent pas de définir ou de différencier un commentaire toxique d'un commentaire non toxique.

4.1.4. Suppression des termes non pertinents

En conséquence, nous avons décidé de **supprimer ces mots** des données avant l'entraînement du modèle. Cette étape est cruciale pour éviter que le modèle n'accorde une importance injustifiée à des termes qui n'ont pas de lien direct avec la toxicité d'une phrase. La suppression a été implémentée à l'aide de filtres textuels automatisés, garantissant une application cohérente sur l'ensemble du dataset.

4.1.5. Impact du prétraitement

Figure 13 : Top 20 des mots les plus fréquents après prétraitement

Ce processus de prétraitement a permis de :

- **Éliminer les biais potentiels** liés à la surreprésentation de certains mots dans les deux classes de labels.
- **Renforcer la pertinence des données** utilisées pour l'entraînement du modèle, en focalisant sur les termes véritablement indicateurs de toxicité.
- **Optimiser les performances du modèle**, en minimisant les erreurs causées par des variables inutiles ou redondantes.

Cette étape constitue une base solide pour l'entraînement du modèle et démontre l'importance d'un nettoyage approfondi dans tout projet de traitement du langage naturel.

4.2. Classification avec LSTM

4.2.1. Architecture du modèle

L'architecture du modèle que nous avons mise en place repose sur un **modèle LSTM (Long Short-Term Memory)** pour la classification binaire de commentaires en fonction de leur toxicité. Une explication détaillée de l'architecture du modèle est la suivante :

- Le prétraitement du texte est essentiel pour transformer les commentaires en une forme compréhensible par le modèle. Cela implique de nettoyer le texte (mettre en minuscules, supprimer la ponctuation et les chiffres), de créer un vocabulaire basé sur les mots les plus fréquents et de convertir chaque commentaire en une séquence d'indices correspondant à des mots dans ce vocabulaire, avec une longueur uniforme. La classe CommentDataset, qui est une sous-classe de torch.utils.data.Dataset, gère les données d'entraînement et de validation en appliquant ce prétraitement sur les textes et en transformant les labels en tenseurs PyTorch, prêts à être utilisés dans la fonction de perte. Cela permet de réduire la variabilité des données et de préparer efficacement les entrées pour l'entraînement du modèle ;
- Le modèle LSTMClassifier est un réseau de neurones récurrent basé sur la variante LSTM, conçu pour traiter des séquences de données, comme les commentaires texte. Il se compose d'une couche d'embedding qui transforme les indices des mots en vecteurs denses, d'une couche LSTM capable de conserver des informations à long terme grâce à ses paramètres comme embedding_dim, hidden_dim, et n_layers, permettant de capturer des patterns complexes dans les séquences. Un mécanisme de régularisation, le **dropout**, est appliqué pour éviter le sur-apprentissage. Enfin, une couche entièrement connectée (fully connected) prend la dernière sortie cachée de l'LSTM pour prédire une des deux classes (toxique ou non toxique). Le modèle inclut également un dropout sur les états cachés pour renforcer la régularisation pendant l'entraînement ;
- La fonction d'entraînement, train_model, entraîne le modèle sur un jeu de données d'entraînement pendant plusieurs époques en utilisant la perte de cross-entropie et l'optimiseur Adam. À chaque époque, la performance du modèle est évaluée sur un jeu de validation pour suivre l'évolution de la perte et de la précision. Ces métriques sont calculées pour les ensembles d'entraînement et de validation, et des graphiques sont générés pour visualiser leur progression. La fonction evaluate_model, quant à elle, est utilisée après l'entraînement pour évaluer les performances du modèle sur un jeu de test ou de validation, en comparant les prédictions aux véritables étiquettes et en retournant les résultats sous forme de tableaux de prédictions et de valeurs réelles ;

- L'architecture du modèle repose sur une approche robuste et fiable pour prédire la toxicité des commentaires. Tout d'abord, les données textuelles sont prétraitées pour être converties en une forme numérique compréhensible par le modèle. Ensuite, un modèle LSTM (Long Short-Term Memory) est utilisé pour saisir les relations séquentielles présentes dans les commentaires. L'entraînement du modèle s'effectue avec l'optimiseur Adam et la fonction de perte CrossEntropyLoss. Afin d'évaluer les performances du modèle de manière fiable, une validation croisée à 5 plis (StratifiedKFold) est mise en place, garantissant que le modèle est testé sur différentes sous-parties des données. Après chaque itération de validation croisée, le modèle est sauvegardé avec son état et celui de l'optimiseur, permettant ainsi de le réutiliser sans avoir à le réentraîner. Cette approche combine des techniques avancées de traitement du langage naturel avec des réseaux de neurones pour prédire efficacement la toxicité des commentaires tout en garantissant la robustesse et la fiabilité des résultats.
-

4.2.2. Explication des résultats

L'analyse des résultats de la classification des commentaires avec le modèle LSTM (Long Short-Term Memory) montre plusieurs points clés concernant les performances et les limitations du modèle dans cette tâche. Ainsi nous avons dressé le tableau pour consigner les résultats issus de l’entrainement du modèles LSTM.

Tableau 7 : Les différentes métriques obtenues à la suite de l'entraînement du modèle LSTM

Fold	**Epoque**	**LSTM**			
		Training Loss	Validation Loss	Training Accuracy	Validation Accuracy
1	1	0.6882	0.6867	0.5591	0.5584
	2	0.6874	0.6868	0.5602	0.5584
	3	0.6870	0.6873	**0.5605**	0.5584
	4	0.6845	0.6869	0.5602	0.5584
	5	0.6850	0.6868	0.5602	**0.5584**
2	1	0.6882	0.6866	0.5552	0.5598
	2	0.6876	0.6865	0.5598	0.5598
	3	0.6892	0.6897	0.5463	0.5556
	4	0.6871	0.6881	**0.5602**	0.5598
	5	0.6874	0.6861	0.5598	**0.5598**
3	1	0.6876	0.6871	0.5580	0.5598
	2	0.6877	0.6862	0.5598	0.5598
	3	0.6870	0.6873	**0.5605**	0.5584
	4	0.6866	0.6856	0.5598	0.5598
	5	0.6859	0.6848	0.5598	**0.5598**
4	1	0.6871	0.6872	0.5588	0.5598
	2	0.6876	0.6864	0.5591	0.5598
	3	0.6871	0.6865	**0.5598**	0.5584
	4	0.6853	0.6601	0.5580	0.5598
	5	0.6901	0.6667	0.5581	**0.5598**
5	1	0.6888	0.6866	0.5566	0.5598
	2	0.6857	0.6941	**0.5602**	0.4387
	3	0.6879	0.6876	0.5509	0.5598
	4	0.6853	0.6864	0.5595	0.5598
	5	0.6853	0.6862	0.5595	**0.5598**

L'analyse des résultats du modèle LSTM montre :

- La précision du modèle reste stable autour de 55-56% pour les différents plis de validation, tant pendant l'entraînement que pendant la validation, ce qui suggère que le modèle a du mal à apprendre des distinctions significatives entre les classes ou qu'il surajuste sans saisir les patterns importants. La perte d'entraînement varie entre 0.684 et 0.690, avec une légère diminution, indiquant un apprentissage lent mais constant. Cependant, la perte de validation reste également stable avec de légères améliorations, sans changement significatif en termes de convergence, ce qui pourrait indiquer un problème de sur apprentissage ou une incapacité à généraliser correctement ;
- Les problèmes potentiels incluent un déséquilibre ou du bruit dans les données, ce qui complique la séparation entre les classes, ainsi que des hyperparamètres qui pourraient nécessiter un ajustement, comme le taux d'apprentissage, les dimensions d'embedding et les dimensions cachées. Le modèle LSTM utilisé pourrait ne pas être suffisamment performant pour cette tâche spécifique, et des architectures plus complexes, telles que des LSTM bidirectionnels ou des transformers, pourraient être plus adaptées. Pour améliorer les résultats, il serait utile d'ajuster les hyperparamètres, d'augmenter le nombre d'époques pour permettre au modèle de converger davantage et d'explorer des techniques de régularisation pour éviter le surajustement.

4.3. Classification avec BERT

4.3.1. Architecture du modèle

L'architecture du modèle dans ce code est basée sur BERT (Bidirectional Encoder Representations from Transformers) pour la classification de texte, combiné avec une approche de validation croisée stratifiée. L’architecture du modèle est la suivante :

- BERT est un modèle de transformer pré-entraîné qui capte les dépendances contextuelles entre les mots, en utilisant la version bert-base-uncased pour encoder chaque commentaire et comprendre à la fois le contexte local et global des mots. Pour la tâche de classification des commentaires, la classe AutoModelForSequenceClassification est utilisée, ajustant la sortie de BERT pour générer une prédiction de classe, indiquant si le commentaire est toxique ou non. Les données textuelles sont d'abord tokenisées à l'aide du tokenizer BERT (AutoTokenizer.from_pretrained("bert-base-uncased")), convertissant le texte brut en séquences d'ID de tokens, ajustées à une longueur maximale de 128 tokens. Chaque

entrée est également accompagnée d'un masque d'attention pour indiquer les parties du texte à prendre en compte par BERT ;

- Le Stratified K-Fold est utilisé pour diviser les données en plusieurs sous-ensembles équilibrés en termes de classes, garantissant ainsi une évaluation robuste du modèle et minimisant les biais. Le modèle est entraîné avec la fonction de perte CrossEntropyLoss, idéale pour les tâches de classification, et l'optimiseur Adam ajuste les paramètres du modèle. Les performances sont évaluées à chaque époque sur l'ensemble d'entraînement et de validation en termes de perte et de précision. Après chaque pli de validation croisée, le modèle est sauvegardé, et des graphiques sont générés pour visualiser l'évolution des pertes et des précisions au cours des époques, offrant ainsi une analyse approfondie de l'apprentissage du modèle.

4.3.2. Explication des résultats

L'analyse des résultats obtenus lors de l'exécution de la validation croisée avec le modèle BERT pour la classification de textes (par exemple, détection de commentaires toxiques) peut être réalisée à partir des pertes et des précisions pour chaque fold. Ainsi nous avons dressé le tableau pour consigner les résultats issus de l’entrainement du modèles BERT.

Tableau 8 : Les différentes métriques obtenues à la suite de l'entraînement du modèle BERT

Fold	**Epoque**	**BERT**			
		Training Loss	Validation Loss	Training Accuracy	Validation Accuracy
1	1	0.6095	0.5332	0.6734	0.7208
	2	0.4698	0.4590	0.7796	0.7920
	3	0.3246	0.4182	0.8732	**0.8276**
	4	0.2368	0.5069	0.9078	0.8020
	5	0.1789	0.5561	**0.9306**	0.8191
2	1	0.6232	0.5961	0.6371	0.7151
	2	0.4593	0.4641	0.7831	0.7920
	3	0.3046	0.5027	0.8782	0.7863
	4	0.2082	0.6204	0.9177	**0.7949**
	5	0.1516	0.7731	**0.9441**	0.7635
3	1	0.6091	0.5521	0.6681	0.7165
	2	0.4413	0.5323	0.7949	0.7835
	3	0.2983	0.5065	0.8843	**0.7963**
	4	0.2212	0.5265	0.9135	0.7963
	5	0.1613	0.5235	**0.9412**	0.7963
4	1	0.6139	0.5155	0.6635	0.7493
	2	0.4633	0.4641	0.7899	0.7721
	3	0.3265	0.4712	0.8675	0.8048
	4	0.2269	0.5354	0.9106	**0.8162**
	5	0.1668	0.5855	**0.9355**	0.7735
5	1	0.6298	0.5143	0.6371	0.7536
	2	0.4435	0.4270	0.8070	0.8105
	3	0.2915	0.4557	0.8796	0.8034
	4	0.1871	0.4954	0.9277	**0.8234**
	5	0.1388	0.5126	**0.9491**	0.7963

L'analyse des résultats du modèle BERT montre :

- Les performances du modèle semblent relativement constantes à travers les différents folds, bien qu'il y ait quelques variations notables dans la perte et l'accuracy. Le **Training Loss** diminue régulièrement, ce qui montre que le modèle apprend et s'adapte

aux données d'entraînement alors que le **Training Accuracy** augmente continuellement, avec des valeurs finales supérieures à 90% dans tous les folds. Cela montre que le modèle arrive à bien s'ajuster aux données d'entraînement ;

- Les **Validation Loss** montrent des tendances variables, dans certains folds, la perte de validation augmente après quelques époques, tandis que dans d'autres, elle continue de diminuer. Cette variation pourrait être due à des fluctuations naturelles ou à des problèmes d'overfitting. Le fait que la perte de validation n'augmente pas de manière significative dans la majorité des cas est un bon signe, mais des ajustements peuvent être nécessaires pour éviter un sur apprentissage ;
- L'**Accuracy de validation** varie entre 76% et 83%, ce qui indique une performance relativement stable sur les ensembles de validation à travers les folds. Cependant, il y a des fluctuations intéressantes, par exemple, dans le Fold 1 où l'accuracy de validation atteint 82%, mais diminue à 77% dans le Fold 4. Ces différences peuvent être dues à des variations dans la répartition des données entre les plis.

Enfin, les visualisations des courbes de **perte** et **précision** montrent des tendances intéressantes pour chaque fold, ce qui peut être utilisé pour affiner davantage les hyper paramètres du modèle.

4.4. Classification Mixte en utilisant Bert et LSTM

4.4.1. Architecture du modèle

Le modèle de classification des commentaires combine BERT, un modèle transformer puissant pour les représentations de texte, avec un LSTM bidirectionnel pour capturer les dépendances à long terme dans les séquences textuelles. Cette architecture est suivie d'une couche fully connected pour effectuer la classification. L'architecture BERT combiné avec du LSTM est particulièrement adaptée pour les tâches complexes de classification de texte, en tirant parti des avantages des deux modèles pour comprendre à la fois le contexte global et les relations temporelles dans les données textuelles. L’architecture du modèle est la suivante :

- Le modèle commence par passer les commentaires à travers BERT pour obtenir des embeddings, qui capturent le sens profond du texte et les relations complexes entre les mots. Les sorties de BERT sont ensuite envoyées à un LSTM bidirectionnel, qui traite les séquences de texte dans les deux directions (gauche à droite et droite à gauche), permettant de mieux saisir le contexte global. Les sorties cachées des deux directions sont concaténées pour capturer l'information des deux sens. Enfin, cette représentation

concaténée est envoyée à une couche fully connected (FC), qui génère la prédiction finale du modèle, en classifiant les commentaires comme toxiques ou non toxiques ;

- Le but du modèle est de classer les commentaires en fonction de leur toxicité ou d'autres catégories. Il s'agit donc d'un modèle de classification supervisée avec des labels binaires (toxique/non toxique ou toute autre tâche de classification) ;
- Le modèle est entraîné en utilisant la fonction de perte CrossEntropyLoss et l'optimiseur Adam pour minimiser cette perte pendant l'entraînement. Après chaque époque, les performances sont évaluées sur un ensemble de validation, avec un suivi de la perte et de la précision pour mesurer la capacité du modèle à généraliser. La validation croisée stratifiée (K-Fold) est utilisée pour évaluer la performance, où les données sont divisées en plusieurs plis. Le modèle est formé sur certains plis et testé sur d'autres, ce qui permet d'assurer sa robustesse et d'éviter les biais liés à la partition des données.

4.4.2. Explication des résultats

L'analyse des résultats du modèle de classification mixte utilisant **BERT** et **LSTM** à travers les différents plis (folds) de validation croisée montre plusieurs informations intéressantes sur les performances du modèle au fil des époques. Ainsi nous avons dressé le tableau pour consigner les résultats issus de l'entrainement du modèles mixte LSTM et BERT.

Tableau 9 : Les différentes métriques obtenues à la suite de l'entraînement du modèle mixte LSTM et BERT

Fold	**Epoque**	**LSTM + BERT**			
		Training Loss	Validation Loss	Training Accuracy	Validation Accuracy
1	1	0.6094	0.5289	0.6660	0.7194
	2	0.4412	0.4434	0.7917	0.7934
	3	0.2754	0.5517	0.8860	0.7650
	4	0.1886	0.4727	0.9291	0.8205
	5	0.1220	0.6400	**0.9548**	**0.8219**
2	1	0.6033	0.5265	0.6717	0.7536
	2	0.4385	0.4544	0.7927	0.7934
	3	0.2703	0.5211	0.8885	0.8077
	4	0.1768	0.6116	0.9306	0.7835
	5	0.1309	0.5403	**0.9480**	**0.8191**
3	1	0.6074	0.5448	0.6581	0.7279
	2	0.4266	0.4716	0.8066	0.7821
	3	0.2828	0.5517	0.8807	0.7778
	4	0.1797	0.6287	0.9284	0.7778
	5	0.1310	0.7167	**0.9562**	**0.7906**
4	1	0.6073	0.5508	0.6738	0.7251
	2	0.4348	0.4418	0.8793	0.7792
	3	0.2872	0.4877	0.8793	0.7792
	4	0.1823	0.5365	0.9320	**0.8077**
	5	0.1302	0.5699	**0.9512**	0.7877
5	1	0.6173	0.5101	0.6613	0.7792
	2	0.4426	0.4471	0.8052	0.7949
	3	0.2793	0.4547	0.8896	**0.8162**
	4	0.1728	0.5352	0.9366	0.8105
	5	0.1288	0.6378	**0.9519**	0.7949

L'analyse des résultats du modèle mixte LSTM et BERT montre :

- Le modèle montre une tendance générale à réduire les pertes d'entraînement au fil des époques, ce qui reflète une bonne optimisation, avec des pertes d'entraînement passant de 60,94% à 12,20% dans le Fold 1. Cependant, les pertes de validation fluctuent et peuvent indiquer un léger sur apprentissage, comme on le voit au Fold 1 où la perte de validation augmente légèrement de 52,89% à 64%. En termes de précision, le modèle améliore continuellement sa précision d'entraînement, atteignant 95.48% au Fold 1, ce qui montre une bonne adaptation aux données d'entraînement. La précision de validation fluctue également entre 72% et 82%, signalant que bien que le modèle généralise correctement dans la plupart des plis ;
- Les performances du modèle varient légèrement entre les différents plis, ce qui est attendu en validation croisée. Le Fold 1 montre la meilleure performance avec une précision de validation de 82.19%, tandis que le Fold 2 atteint 81.91%, avec une légère instabilité à la dernière époque. Le Fold 3 présente une précision de validation de 79.06%, mais avec une perte de validation plus élevée (71,67%). Le Fold 4 reste stable avec une précision de 78.77%, et le Fold 5 conserve des résultats compétitifs avec 79.49%, malgré des fluctuations dans les pertes de validation. En somme, les précisions de validation varient entre 78% et 82%, montrant des résultats relativement constants ;
- Une tendance au sur apprentissage est observée, notamment dans les plis où la perte de validation devient plus instable, comme au Fold 3, où elle augmente de manière significative (de 55,68% à 71,67%). Cependant, la précision reste élevée, ce qui montre que le modèle généralise encore bien sur les données de validation ;
- Les courbes de **perte** et **précision** montrent une bonne convergence de l'entraînement pour la majorité des plis, mais avec quelques fluctuations dans les pertes de validation. Les courbes de précision d'entraînement montrent un modèle qui devient de plus en plus performant, mais les courbes de précision de validation montrent que, bien que la précision soit généralement élevée, elle pourrait bénéficier d'un ajustement ;
- Les résultats de la combinaison du modèle montrent que BERT et LSTM fonctionne bien pour la classification de textes, avec des précisions de validation entre 79% et 82%. Cependant, l'instabilité de la perte de validation suggère un risque de sur apprentissage. Il serait bénéfique d'explorer des techniques comme le dropout ou d'ajuster les hyper paramètres pour améliorer la généralisation du modèle. Bien que la combinaison du modèle BERT et LSTM soit efficace, des optimisations supplémentaires pourraient améliorer encore les performances.

4.5. Comparaison des résultats et Synthèse

4.5.1. Résultats des modèles

Tableau 10 : Les différentes métriques obtenues à la suite de l'entraînement des différents modèles

Fold	Epoque	LSTM				BERT				LSTM + BERT			
		Training Loss	Validation Loss	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss	Training Accuracy	Validation Accuracy
1	1	0.6882	0.6867	0.5591	0.5584	0.6095	0.5332	0.6734	0.7208	0.6094	0.5289	0.6660	0.7194
	2	0.6874	0.6868	0.5602	0.5584	0.4698	0.4590	0.7796	0.7920	0.4412	0.4434	0.7917	0.7934
	3	0.6870	0.6873	**0.5605**	**0.5584**	0.3246	0.4182	0.8732	**0.8276**	0.2754	0.5517	0.8860	0.7650
	4	0.6845	0.6869	0.5602	0.5584	0.2368	0.5069	0.9078	0.8020	0.1886	0.4727	0.9291	0.8205
	5	0.6850	0.6868	0.5602	0.5584	0.1789	0.5561	0.9306	0.8191	0.1220	0.6400	0.9548	**0.8219**
2	1	0.6882	0.6866	0.5552	0.5598	0.6232	0.5961	0.6371	0.7151	0.6033	0.5265	0.6717	0.7536
	2	0.6876	0.6865	0.5598	0.5598	0.4593	0.4641	0.7831	0.7920	0.4385	0.4544	0.7927	0.7934
	3	0.6892	0.6897	0.5463	0.5556	0.3046	0.5027	0.8782	0.7863	0.2703	0.5211	0.8885	0.8077
	4	0.6871	0.6881	**0.5602**	**0.5598**	0.2082	0.6204	0.9177	0.7949	0.1768	0.6116	0.9306	0.7835
	5	0.6874	0.6861	0.5598	0.5598	0.1516	0.7731	0.9441	0.7635	0.1309	0.5403	0.9480	0.8191
3	1	0.6876	0.6871	0.5580	0.5598	0.6091	0.5521	0.6681	0.7165	0.6074	0.5448	0.6581	0.7279
	2	0.6877	0.6862	0.5598	0.5598	0.4413	0.5323	0.7949	0.7835	0.4266	0.4716	0.8066	0.7821
	3	0.6870	0.6873	**0.5605**	0.5584	0.2983	0.5065	0.8843	0.7963	0.2828	0.5517	0.8807	0.7778
	4	0.6866	0.6856	0.5598	0.5598	0.2212	0.5265	0.9135	0.7963	0.1797	0.6287	0.9284	0.7778
	5	0.6859	0.6848	0.5598	0.5598	0.1613	0.5235	0.9412	0.7963	0.1310	0.7167	**0.9562**	0.7906
4	1	0.6871	0.6872	0.5588	0.5598	0.6139	0.5155	0.6635	0.7493	0.6073	0.5508	0.6738	0.7251
	2	0.6876	0.6864	0.5591	0.5598	0.4633	0.4641	0.7899	0.7721	0.4348	0.4418	0.8793	0.7792
	3	0.6871	0.6865	0.5598	0.5584	0.3265	0.4712	0.8675	0.8048	0.2872	0.4877	0.8793	0.7792
	4	0.6853	0.6601	0.5580	0.5598	0.2269	0.5354	0.9106	0.8162	0.1823	0.5365	0.9320	0.8077
	5	0.6901	0.6667	0.5581	0.5598	0.1668	0.5855	0.9355	0.7735	0.1302	0.5699	0.9512	0.7877
5	1	0.6888	0.6866	0.5566	0.5598	0.6298	0.5143	0.6371	0.7536	0.6173	0.5101	0.6613	0.7792
	2	0.6857	0.6941	0.5602	0.4387	0.4435	0.4270	0.8070	0.8105	0.4426	0.4471	0.8052	0.7949
	3	0.6879	0.6876	0.5509	0.5598	0.2915	0.4557	0.8796	0.8034	0.2793	0.4547	0.8896	0.8162
	4	0.6853	0.6864	0.5595	0.5598	0.1871	0.4954	0.9277	0.8234	0.1728	0.5352	0.9366	0.8105
	5	0.6853	0.6862	0.5595	0.5598	0.1388	0.5126	**0.9491**	0.7963	0.1288	0.6378	0.9519	0.7949

4.5.2. Graphe de comparaison des modèles

Nous pouvons représenter le graphique comparant les performances des modèles LSTM, BERT et LSTM+BERT en termes de précision (Accuracy) et de perte (Loss) sur les ensembles d'entraînement et de validation, à travers différentes époques et plis.

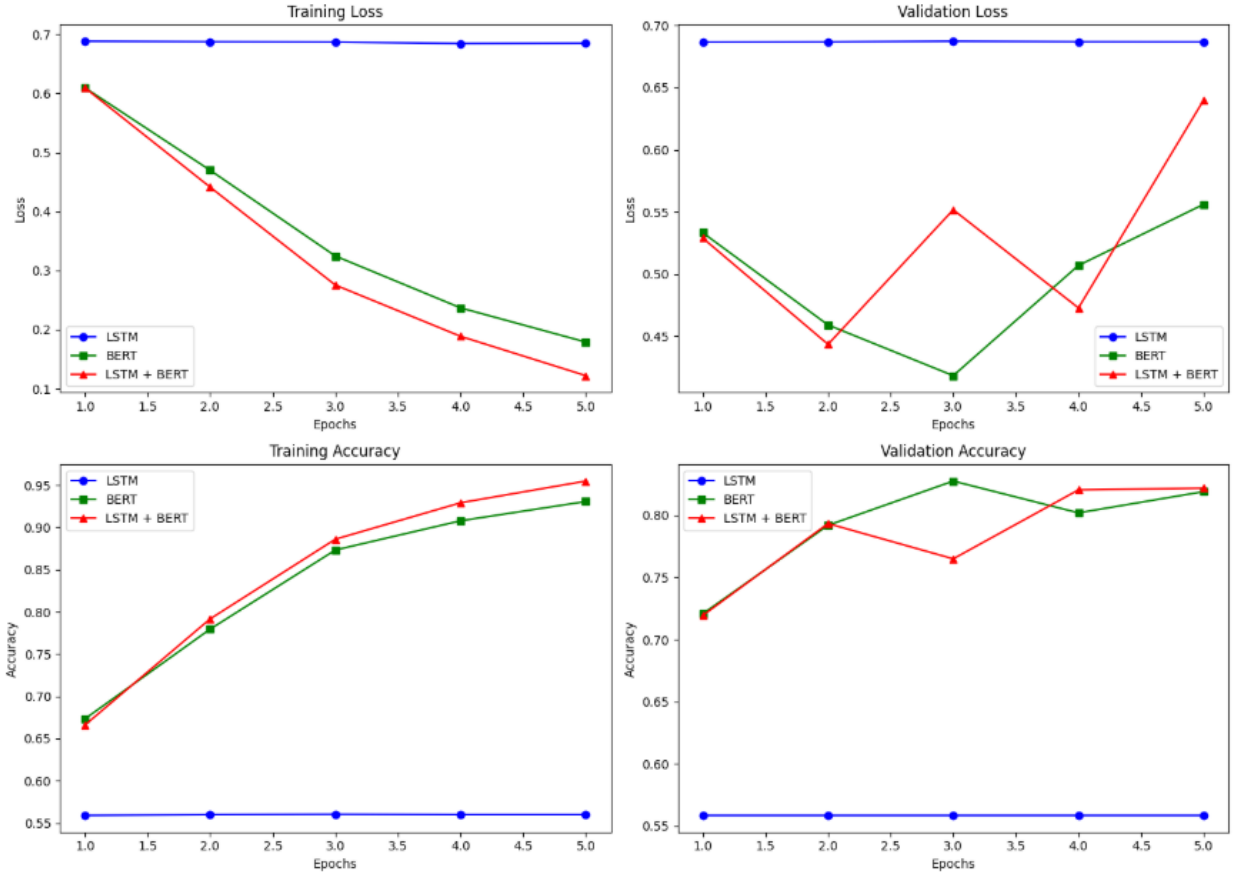

Figure 14 : Comparaison des performances de LSTM, BERT et LSTM+BERT

4.5.3. Moyennes des métriques par Fold

Tableau 11 : Les différentes moyennes des métriques obtenues à la suite de l'entraînement des différents modèles

Fold	Moyennes	LSTM				BERT				LSTM + BERT			
		Training Loss	Validation Loss	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss	Training Accuracy	Validation Accuracy
1	**Moyennes**	0.6854	0.6865	0.5600	0.5583	0.4448	0.5147	0.7852	0.7838	0.4423	0.5046	0.7868	0.7833
2	**Moyennes**	0.6854	0.6863	0.5602	0.5596	0.4569	0.5064	0.7925	0.7824	0.4537	0.5059	0.7927	0.7833
3	**Moyennes**	0.6851	0.6860	0.5603	0.5588	0.4494	0.5023	0.7925	0.7836	0.4463	0.5027	0.7925	0.7837
4	**Moyennes**	0.6852	0.6865	0.5601	0.5597	0.4613	0.5071	0.7922	0.7823	0.4570	0.5085	0.7925	0.7834
5	**Moyennes**	0.6852	0.6863	0.5602	0.5597	0.4543	0.5105	0.7923	0.7832	0.4513	0.5103	0.7925	0.7835

Une analyse globale du tableau des moyennes et du graphe montre que :

- **Le modèle LSTM** a montré une très faible performance comparée à BERT, avec une précision de validation faible et des pertes élevées. Ce modèle pourrait ne pas être suffisamment complexe pour capturer les structures profondes du langage ;
- **Le modèle BERT** semble être le modèle le plus performant avec la meilleure précision de validation d'environ 80%. Il a également la capacité d'exploiter les représentations contextuelles profondes du texte, ce qui en fait un choix solide pour ce type de tâche de classification de texte ;
- **Le modèle LSTM combiné au BERT** présente une amélioration par rapport à LSTM seul, mais ne surpasse pas de manière significative BERT seul en termes de performances. Cependant, cela pourrait être bénéfique dans des contextes spécifiques où l'ajout de LSTM améliore la gestion de séquences.

4.5.4. Test sur les folds

Mais au-delà de l'analyse que nous avons fait sur les différents modèles après leur entrainement nous pouvons encore faire une analyse supplémentaire sur les résultats des folds sauvegardés. A travers l'observation de la précision, du Rcall et du F1_Score des différents modèles cela nous permet de dresser le tableau suivant.

Tableau 12 : Precision, Rcall et F1_Score des différentes folds de chaque modèle

FOLD	LSTM			BERT			LSTM + BERT		
	Precision	Recall	F1_Score	Precision	Recall	F1_Score	Précision	Recall	F1_Score
1	0.31	0.56	**0.40**	0.97	0.92	**0.94**	0.88	0.78	0.83
2	0.75	0.55	0.40	0.85	0.96	0.90	0.94	0.92	0.93
3	0.60	0.55	0.40	0.95	0.92	0.93	0.96	0.98	**0.97**
4	0.75	0.55	0.40	0.88	0.95	0.92	0.95	0.98	0.96
5	0.31	0.56	0.40	0.90	0.92	0.91	0.95	0.97	0.96

Les résultats des modèles LSTM, BERT et LSTM + BERT montrent des performances variées. LSTM présente des scores de précision modérés et un faible F1-Score, avec un rappel stable mais faible. BERT, en revanche, excelle avec une très haute précision, un rappel élevé et un F1-Score élevé, indiquant une excellente capacité de classification. La combinaison LSTM + BERT montre des résultats encore meilleurs, avec des scores de précision et de rappel élevés et un F1-Score supérieur, offrant un équilibre optimal entre ces métriques. Bien que BERT seul soit performant, LSTM + BERT semble être le choix le plus adapté pour obtenir une performance globale optimale.

4.6. Discussion

Les résultats obtenus avec les nouvelles données de test, pour nos deux modèles, qui sont les plus performants sur notre dataset après l'entraînement, sont consignés dans le tableau suivant :

Tableau 13 : Precision, Rcall et F1_Score avec de nouvelles données de test

BERT			**LSTM + BERT**		
Precision	Recall	F1_Score	Précision	Recall	F1_Score
50%	90,91%	64,52%	56,25%	81,82%	66,67%

Après l'entraînement des modèles BERT et LSTM + BERT, une analyse des résultats obtenus sur les données de test révèle que BERT, bien qu'ayant un rappel élevé de 90,91%, souffre d'une précision faible (50%), ce qui signifie qu'il détecte de nombreux commentaires toxiques mais commet beaucoup de faux positifs. En revanche, le modèle LSTM + BERT améliore la précision à 56,25% et le F1-Score à 66,67%, tout en ayant un rappel légèrement inférieur (81,82%), suggérant une meilleure gestion des faux positifs tout en maintenant une couverture solide des commentaires toxiques. En résumé, LSTM + BERT semble être légèrement supérieur à BERT seul, avec un compromis favorable entre la précision et le rappel. Cependant, des ajustements dans l'architecture du modèle ou les stratégies d'entraînement peuvent encore améliorer les performances de manière significative.

4.7. Présentation de la solution

Figure 15 : Interface pour tester des commentaires

L’interface que nous avons conçue permet de tester des commentaires afin d'évaluer ou de prédire s'ils sont toxiques ou non toxiques. L'interface se compose de quatre éléments principaux :

1 **Champ de saisie** : Le champ intitulé "Entrez votre commentaire :" permet à l'utilisateur de saisir un commentaire pour l'analyser ;

2 **Bouton "Tester"** : Ce bouton permet de soumettre le commentaire saisi à notre algorithme d'analyse. Une fois activé, il déclenche le processus de prédiction pour déterminer si le commentaire est toxique ou non toxique ;

3 **Bouton "Effacer"** : Ce bouton permet de vider le contenu du champ de saisie, offrant ainsi une nouvelle opportunité de test sans avoir à supprimer manuellement le texte ;

4 **Section "Résultat de la prédiction"** : Cette section comprend deux champs qui affichent les résultats de l'analyse. Le premier champ indique la probabilité que le commentaire soit toxique, tandis que le second champ montre la probabilité que le commentaire soit non toxique. Ces résultats permettent une évaluation claire et détaillée de la nature du commentaire.

Cette interface intuitive et facile à utiliser est conçue pour aider les utilisateurs à évaluer rapidement la nature des commentaires, ce qui peut être particulièrement utile pour les modérateurs de contenu ou les plateformes en ligne cherchant à maintenir un environnement respectueux et sûr.

Conclusion et perspectives

Conclusion

Dans ce mémoire nous avons travail sur la modération des commentaires toxiques sur les réseaux sociaux par machine learning au Sénégal, cette modération est devenue un enjeu majeur dans le maintien de la sécurité et de la convivialité des espaces numériques. En particulier, dans un pays comme le Sénégal, où la diversité linguistique et culturelle joue un rôle crucial dans les interactions en ligne, il est impératif de développer des systèmes de modération adaptés aux spécificités locales. Cette étude, centrée sur l'utilisation des techniques de machine learning pour détecter et modérer les commentaires toxiques, vise à répondre aux défis posés par cette problématique. À travers une approche qualitative et l'adaptation des algorithmes de machine learning, nous avons exploré comment les technologies modernes peuvent améliorer l'efficacité des systèmes de modération. Nous avons démontré que la combinaison de méthodes telles que les réseaux neuronaux, la classification de texte, et les modèles pré-entraînés comme BERT peut apporter des solutions pertinentes, même dans un contexte linguistique complexe comme celui du Sénégal, où les commentaires sont souvent multilingues et mélangent des expressions locales avec des termes empruntés à d'autres langues.

Notre objectif principal était de clarifier plusieurs concepts clés liés à la modération des propos toxiques, et a mis en lumière les défis spécifiques liés à la diversité culturelle et linguistique. Nous avons ainsi identifié que l'adaptation des outils de machine learning au contexte sénégalais, en prenant en compte la langue Wolof et d'autres variations locales, améliore considérablement la précision de la détection des commentaires toxiques. De plus, l'utilisation de la validation croisée et d'une évaluation minutieuse des performances des modèles a permis de valider l'efficacité des algorithmes proposés, tout en soulignant les zones d'amélioration possibles. L'approche qualitative adoptée a également révélé l'importance d'une analyse contextuelle dans la modération des commentaires. En effet, la toxicité d'un commentaire peut être subtile et dépend de nombreux facteurs contextuels, tels que le ton employé, l'intention derrière les mots, ou encore les interactions avec d'autres utilisateurs. Ces éléments sont souvent invisibles dans les approches purement quantitatives, et c'est là qu'une modération intelligente, alimentée par des modèles d'intelligence artificielle avancée, devient cruciale pour garantir une modération non seulement automatique, mais aussi pertinente.

Nous avons proposé l'adaptation des algorithmes aux spécificités locales, en particulier la prise en charge des langues mixtes, pourrait être une étape clé pour améliorer encore la précision et l'efficacité de ces systèmes. En termes d'applications pratiques, cette étude montre qu'il est

possible d'implémenter un système de modération des commentaires toxiques capable de détecter non seulement des propos injurieux, mais aussi des discours haineux, discriminatoires ou autres formes de toxicité. Cela permet non seulement de préserver un espace d'expression libre et respectueux pour les utilisateurs, mais aussi de soutenir une gouvernance des contenus en ligne, renforçant ainsi la confiance des citoyens dans l'espace numérique. Les résultats obtenus à travers l'application du machine learning dans cette étude ouvrent la voie à une modération plus efficace et proactive des commentaires sur les plateformes en ligne au Sénégal. En outre, dans les perspectives cette étude propose une méthodologie qui pourrait être étendue à d'autres pays et contextes, où la diversité linguistique et culturelle exige des solutions personnalisées. Enfin, l'impact de cette recherche dépasse le cadre académique, en offrant une solution concrète aux autorités locales, aux entreprises et aux plateformes en ligne pour renforcer la sécurité numérique et la convivialité sur internet. La mise en œuvre de systèmes de modération basée sur le machine learning peut contribuer à créer un environnement numérique plus sain, tout en préservant la liberté d'expression, un principe fondamental dans les sociétés modernes. Ainsi, cette étude représente une contribution significative à la compréhension et à la mise en place de systèmes de modération adaptés aux réalités culturelles et linguistiques du Sénégal, et constitue une avancée dans la lutte contre les propos toxiques sur les réseaux sociaux à l'échelle mondiale.

Références

1. Abiteboul, S., & Cattan, J. (2020). Nos réseaux sociaux, notre régulation. *Red*, *1*(1), 36-44.
2. ADJANOHOUN, M. J., & GBAGUIDI, J. K. ANALYSE SUBSTANTIELLE DES ÉNONCÉS INSULTANTS EN FƆNGBÈ. PDF disponible https://www.revue-akofena.com/wp-content/uploads/2021/09/28-T03-37-pp.-371-388.pdf
3. Sierra-Scroccaro, N. (2023). Fiche 3 : Le harcèlement sexuel en ligne. *Fiches de psycho*, 31-37.
4. Fortier-Landry, F. (2015). La diffamation sur Internet : Actualiser la responsabilité en droit civil et en common law au Canada. PDF disponible https://papyrus.bib.umontreal.ca/xmlui/handle/1866/11759
5. Baider, F. H. (2019). Le discours de haine dissimulée : le mépris pour humilier. *Déviance et société*, *43*(3), 359-387. PDF disponible https://shs.cairn.info/revue-deviance-et-societe-2019-3-page-359?lang=fr
6. Sanchez Viera, T. (2020). Prédiction de comportements toxiques à partir des messages sur les réseaux sociaux. PDF disponible [1] https://corpus.ulaval.ca/server/api/core/bitstreams/b076458c-3b0a-4727-af3048af25025c67/content
7. Ahmad, B. P. (2019). *Production de ressources multilingues pour l'aide à la traduction du droit pénal en hindi, ourdou et français* (Doctoral dissertation, Institut National des Langues et Civilisations Orientales-INALCO PARIS-LANGUES O'). PDF disponible https://inalco.hal.science/tel-02916259/document
8. Grison, T., Julliard, V., Alié, F., & Ecrement, V. (2023). La modération abusive sur Twitter. *Réseaux*, *237*(1), 119-149.
9. Badouard, R. (2021). Modérer la parole sur les réseaux sociaux : Politiques des plateformes et régulation des contenus. *Réseaux*, (1), 87-120. PDF disponible https://shs.cairn.info/revue-reseaux-2021-1-page-87?lang=fr
10. Faty, L. (2020). Vers un système de fouille d'opinions dans les commentaires de la presse en ligne : cas du Sénégal. PDF disponible http://www.rivieresdusud.uasz.sn:8080/bitstream/handle/123456789/281/lfaty_Th%C3%A8se_Doctorat.pdf?sequence=1&isAllowed=y
11. Chouaib, H. (2011). Sélection de caractéristiques : méthodes et applications. *Paris Descartes University: Paris, France*. PDF disponible
12. DE MATTEIS, L., Janny, S., Nathan, S., & Shu-Quartier, W. (2022). *Introduction à l'apprentissage automatique*.
13. Magnan, C. (2005). Apprentissage semi-supervisé asymétrique et estimations d'affinités locales dans les protéines. In *CAP* (pp. 297-312). PDF disponible https://www.researchgate.net/profile/Christophe-Magnan-2/publication/220718814_Apprentissage_semi-supervise_asymetrique_et_estimations_d'affinites_locales_dans_les_proteines/links/00b49533da367d4502000000/Apprentissage-semi-supervise-asymetrique-et-estimations-daffinites-locales-dans-les-proteines.pdf
14. TOLGUI, H. (2017). Deep Learning pour Reconnaissance du visage. PDF disponible http://archives.univ-biskra.dz/bitstream/123456789/11124/3/TOLGUI-Hocine.pdf
15. Maâmatou, H., Chateau, T., Gazzah, S., Goyat, Y., & Amara, N. E. B. Transfert d'apprentissage par un filtre séquentiel de Monte Carlo : application à la spécialisation d'un détecteur de piétons Transfer Learning by a sequential Monte Carlo filter: application to the specialization of a pedestrian detector. PDF disponible https://orasis2015.sciencesconf.org/56143/Houda_orasis56143.pdf
16. Brassard-Gourdeau, É. (2019). Toxicité et sentiment : comment l'étude des sentiments peut aider la détection de toxicité. PDF disponible : https://corpus.ulaval.ca/server/api/core/bitstreams/6f02a1b3-e91b-4e85-be87-f46728eb88d1/content
17. Audemard, G., Bellart, S., Bounia, L., Lagniez, J. M., Marquis, P., & Szczepanski, N. (2023, January). PyXAI: calculer en Python des explications pour des modèles d'apprentissage

supervisé. In *Extraction et Gestion des Connaissances, EGC*. PDF disponible : https://hal.science/hal-04148656/document

18. Brassard-Gourdeau, É. (2019). Toxicité et sentiment : comment l'étude des sentiments peut aider la détection de toxicité. PDF disponible : https://corpus.ulaval.ca/server/api/core/bitstreams/6f02a1b3-e91b-4e85-be87-f46728eb88d1/content
19. Park, D., Sachar, S., Diakopoulos, N., & Elmqvist, N. (2016, mai). Aider les modérateurs de commentaires à identifier les commentaires d'actualité en ligne de haute qualité. Dans *les actes de la conférence CHI 2016 sur les facteurs humains dans les systèmes informatiques* (pp. 1114-1125). PDF disponible : https://crystal.uta.edu/~park/files/commentiq/commentiq.pdf
20. Boussouf, S. (2024). *La Reconnaissance du Langage Offensant dans le Contenu Arabe en Ligne* (Doctoral dissertation, UNIVERSITY BBA). PDF disponible : https://dspace.univ-bba.dz/xmlui/bitstream/handle/123456789/5660/PFEE.pdf?sequence=1
21. Maus, A. (2009). Approche SVM de la modération des forums et des commentaires. *Projets de classe pour CS*. PDF disponible : https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=144c9d6da8e9fe9b26c6408fdde4efa4533d5c4b
22. Grozavu, N., Bennani, Y., & Lebbah, M. (2009). Caractérisation automatique des classes découvertes en classification non supervisée. In *EGC* (pp. 43-54). PDF disponible : https://www.researchgate.net/profile/Nistor-Grozavu/publication/220786920_Caracterisation_automatique_des_classes_decouvertes_en_classification_non_supervisee/links/0deec51531f67c6543000000/Caracterisation-automatique-des-classes-decouvertes-en-classification-non-supervisee.pdf
23. NASSIROU, D. A. (2023). CATEGORISATION AUTOMATIQUE DU CONTENU OFFENSIF. PDF disponible : https://dspace.univ-guelma.dz/jspui/bitstream/123456789/14815/1/NASSIROU%20DAOUDA_AMINOU_F1.pdf
24. Tessa, B., Cima, L., Trujillo, A., Avvenuti, M., & Cresci, S. (2024). Au-delà des essais et erreurs : prédire l'abandon des utilisateurs après une intervention de modération. *Préimpression arXiv arXiv :2404.14846*. PDF disponible : https://arxiv.org/pdf/2404.14846
25. Petricevic, U. (2023). Regroupement de textes avec des approches simples et efficaces exploitant la représentation vectorielle contextuelle SBERT. PDF disponible : https://papyrus.bib.umontreal.ca/xmlui/bitstream/handle/1866/27952/Petricevic_Uros_2022_Memoire.pdf?sequence=2
26. Gongane, VU, Munot, MV, & Anuse, AD (2022). Détection et modération de contenus préjudiciables sur les plateformes de médias sociaux : état actuel et orientations futures. *Social Network Analysis and Mining*, *12* (1), 129. PDF disponible : https://link.springer.com/content/pdf/10.1007/s13278-022-00951-3.pdf
27. Risch, J., & Krestel, R. (2020). Détection de commentaires toxiques dans les discussions en ligne. *Approches basées sur l'apprentissage profond pour l'analyse des sentiments*, 85-109. PDF disponible : https://hpi.de/fileadmin/user_upload/fachgebiete/naumann/publications/2019/risch2019toxic.pdf
28. dit Avocat, LB, & Riesen, PDK Classification des commentaires toxiques. PDF disponible : https://scholar.google.com/scholar?hl=fr&as_sdt=0%2C5&q=Moderating+Toxic+Online+Comments+Using+Co-training&btnG
29. Frau-Meigs, D. (2019). Chapitre 5. Les réponses en amont : de l'autorégulation à la régulation. *Doc'en poche*, 137-168.
30. Duchêne, C., Jamet, H., Guillaume, P., & Dehak, R. (2023, February). Benchmark pour la classification de commentaires toxiques sur le jeu de données Civil Comments. In *Extraction et Gestion des Connaissances : Actes de la conférence EGC'2023* (Vol. 39). BoD-Books on Demand. PDF disponible https://www.researchgate.net/profile/Corentin-Duchene/publication/367462130_A_benchmark_for_toxic_comment_classification_on_Civil_Comments_dataset/links/63d65c3564fc860638f89728/A-benchmark-for-toxic-comment-classification-on-Civil-Comments-dataset.pdf

31. de Gabriac, C. G. D. F. (2021). *Deep Natural Language Processing for User Representation* (Doctoral dissertation, Sorbonne Université). PDF disponible : https://theses.hal.science/tel-03545621/file/GAINON_Clara_these_2021.pdf
32. Laugier, L. (2022). *Analysis and control of online interactions through neural natural language processing* (Doctoral dissertation, Institut Polytechnique de Paris). PDF disponible https://theses.hal.science/tel-03884481/document
33. Guillaume, P., Duchêne, C., & Dehak, R. (2022). Hate speech and toxic comment detection using transformers. *EPITA Speech and Language Recognition Group (ESLR)*. PDF disponible : https://www.researchgate.net/profile/Corentin-Duchene/publication/360085544_Hate_Speech_and_Toxic_Comment_Detection_using_Transformers/links/62653cab1b747d19c2a36740/Hate-Speech-and-Toxic-Comment-Detection-using-Transformers.pdf
34. Sharifani, K., & Amini, M. (2023). Machine learning and deep learning: A review of methods and applications. *World Information Technology and Engineering Journal*, *10*(07), 3897-3904. PDF disponible : https://papers.ssrn.com/sol3/Delivery.cfm?abstractid=4458723

Table des matières

Annexe 1 : Code pour extraction des commentaires sur des vidéos publiées sur YouTube

```
pip install google-api-python-client pytube

from googleapiclient.discovery import build
import csv
import pandas as pd
import unicodedata
import re


def extraction_commentaires(video_ids, csv_filename='commentaires.csv'):
    # Clé API YouTube (remplace par ta clé API)
    api_key = 'AIzaSyCSLHvzIhUxqFdo6--9wDsSiQZbOHYdym8'

    # Construire le service YouTube
    youtube = build('youtube', 'v3', developerKey=api_key)

    # Fonction pour extraire les commentaires
    def get_comments(youtube, video_id, max_comments=100):
        comments = []
        request = youtube.commentThreads().list(
            part="snippet",
            videoId=video_id,
            maxResults=max_comments,
            textFormat="plainText"
        )
        response = request.execute()

        while request is not None:
            for item in response["items"]:
                comment = item['snippet']['topLevelComment']['snippet']['textDisplay']
                comments.append(comment)

            if 'nextPageToken' in response:
                request = youtube.commentThreads().list(
                    part="snippet",
                    videoId=video_id,
                    pageToken=response['nextPageToken'],
                    maxResults=max_comments,
                    textFormat="plainText"
                )
                response = request.execute()
            else:
                break

        return comments

    # Enregistrer les commentaires de plusieurs vidéos dans un fichier CSV
    with open(csv_filename, 'w', newline='', encoding='utf-8') as csvfile:
        writer = csv.writer(csvfile)
        writer.writerow(["Video ID", "Commentaire"])

        for video_id in video_ids:
            comments = get_comments(youtube, video_id)
            for comment in comments:
                writer.writerow([video_id, comment])

    print(f"Les commentaires de toutes les vidéos ont été enregistrés dans le fichier {csv_filename}")

# Liste des identifiants de vidéos
video_ids = ['lgxSlFiBKvk','DXTKDUzC2kU']

# Appel de la fonction avec plusieurs identifiants de vidéos
extraction_commentaires(video_ids)


nn=pd.read_csv('commentaires.csv')
nn


# Fonction pour nettoyer les commentaires
def clean_comment(comment):
    # Convertir en minuscules
    comment = comment.lower()
    # Remplacer les accents par des lettres normales
    comment = unicodedata.normalize('NFKD', comment).encode('ascii', 'ignore').decode('utf-8')
    # Supprimer les emojis (tous les caractères Unicode de type "symboles et pictogrammes")
    comment = re.sub(r'[\U00010000-\U0010ffff]', '', comment)  # Unicode pour les emojis
    # Supprimer les caractères spéciaux restants (optionnel, garde seulement lettres et espaces)
    comment = re.sub(r'[^\w\s]', '', comment)
    return comment.strip()  # Retirer les espaces en trop au début/fin

# Nettoyer la colonne "Commentaire" et remplacer son contenu
nn['Commentaire'] = nn['Commentaire'].apply(clean_comment)

# Ajouter une nouvelle colonne "label" initialisée à 1
nn['label'] = 0

#nn= nn.drop(columns=['Commentaire'])
nn.to_csv('commentaires_clean.csv', index=False)
df=pd.read_csv("commentaires_clean.csv")
df
```

Annexe 2 : Code pour le prétraitement

styles.css | index.html | Architecture_Extraction_Commentaire.py 9+ | Architecture_Prétraitement.py 6

D: > memoire > Architecture Améliorer > Architecture_Prétraitement.py > ...

```python
import pandas as pd
import matplotlib.pyplot as plt
from wordcloud import WordCloud
from sklearn.feature_extraction.text import CountVectorizer
from nltk.corpus import stopwords
import nltk

# Télécharger les stop words si ce n'est pas déjà fait
nltk.download('stopwords')

# Charger les données
df = pd.read_csv("Données annotées - df_concat V_2.csv", sep=';')
df = df.dropna().reset_index(drop=True)

# Vérifier le contenu
print(df.head())
print(df.info())

df['label'] = df['label'].astype(int)

# Charger les stop words en français
stop_words_french = stopwords.words('french')

# Extraire les commentaires pour la classe 0
class_0_comments = df[df['label'] == 0]['Commentaire']

# Initialiser le CountVectorizer pour la classe 0
vectorizer_0 = CountVectorizer(stop_words=stop_words_french, max_features=20)

# Transformer les textes en vecteurs de caractéristiques
X_0 = vectorizer_0.fit_transform(class_0_comments)

# Extraire les mots les plus fréquents
class_0_words = vectorizer_0.get_feature_names_out()
class_0_counts = X_0.toarray().sum(axis=0)

# Créer un DataFrame pour visualiser les résultats
class_0_freq = pd.DataFrame(zip(class_0_words, class_0_counts), columns=['Word', 'Frequency'])

# Trier par fréquence
class_0_freq = class_0_freq.sort_values(by='Frequency', ascending=False)

print("Mots les plus fréquents pour la classe 0 :")
print(class_0_freq)


# Charger les stop words en français
stop_words_french = stopwords.words('french')

# Extraire les commentaires pour la classe 1
class_1_comments = df[df['label'] == 1]['Commentaire']

# Initialiser le CountVectorizer pour la classe 1
vectorizer_1 = CountVectorizer(stop_words=stop_words_french, max_features=20)

# Transformer les textes en vecteurs de caractéristiques
X_1 = vectorizer_1.fit_transform(class_1_comments)

# Extraire les mots les plus fréquents
class_1_words = vectorizer_1.get_feature_names_out()
class_1_counts = X_1.toarray().sum(axis=0)

# Créer un DataFrame pour visualiser les résultats
class_1_freq = pd.DataFrame(zip(class_1_words, class_1_counts), columns=['Word', 'Frequency'])

# Trier par fréquence
class_1_freq = class_1_freq.sort_values(by='Frequency', ascending=False)

print("Mots les plus fréquents pour la classe 1 :")
print(class_1_freq)

# Générer des nuages de mots
wordcloud_0 = WordCloud(width=800, height=400, background_color='white').generate_from_frequencies(dict(zip(class_0_words, class_0_counts)))
wordcloud_1 = WordCloud(width=800, height=400, background_color='white').generate_from_frequencies(dict(zip(class_1_words, class_1_counts)))

# Afficher les nuages de mots
plt.figure(figsize=(15, 10))
plt.subplot(1, 2, 1)
plt.imshow(wordcloud_0, interpolation='bilinear')
plt.title('Classe 0')
plt.axis('off')

plt.subplot(1, 2, 2)
plt.imshow(wordcloud_1, interpolation='bilinear')
plt.title('Classe 1')
plt.axis('off')

plt.show()
```

```python
import re
# Liste des mots à supprimer
mots_a_supprimer = ['tirailleurs', 'sonko', 'bougane', 'moussa', 'birima', 'birma', 'maitre','diouf']

# Fonction pour supprimer les mots spécifiés dans un commentaire
def supprimer_mots(commentaire):
    if not isinstance(commentaire, str):  # Vérifier si le commentaire est une chaîne de caractères
        return commentaire
    commentaire = commentaire.lower()  # Convertir en minuscule
    pattern = r'\b(' + '|'.join(map(re.escape, mots_a_supprimer)) + r')\b'  # Construire le pattern regex
    commentaire = re.sub(pattern, '', commentaire)  # Supprimer les mots
    commentaire = re.sub(r'\s+', ' ', commentaire).strip()  # Nettoyer les espaces superflus
    return commentaire

# Appliquer la fonction sur tout le DataFrame
df['Commentaire'] = df['Commentaire'].apply(supprimer_mots)

# Sauvegarder les modifications
df.to_csv("Données_nettoyées.csv", index=False)

print("Les mots spécifiques ont été supprimés pour tous les commentaires. Les données nettoyées ont été sauvegardées.")


# Charger les données
df = pd.read_csv("Données_nettoyées.csv", sep=',')
df = df.dropna().reset_index(drop=True)

# Vérifier le contenu
print(df.head())
print(df.info())


df['label'] = df['label'].astype(int)

# Charger les stop words en français
stop_words_french = stopwords.words('french')

# Extraire les commentaires pour la classe 0
class_0_comments = df[df['label'] == 0]['Commentaire']

# Initialiser le CountVectorizer pour la classe 0
vectorizer_0 = CountVectorizer(stop_words=stop_words_french, max_features=20)

# Transformer les textes en vecteurs de caractéristiques
X_0 = vectorizer_0.fit_transform(class_0_comments)

# Extraire les mots les plus fréquents
class_0_words = vectorizer_0.get_feature_names_out()
class_0_counts = X_0.toarray().sum(axis=0)

# Créer un DataFrame pour visualiser les résultats
class_0_freq = pd.DataFrame(zip(class_0_words, class_0_counts), columns=['Word', 'Frequency'])

# Trier par fréquence
class_0_freq = class_0_freq.sort_values(by='Frequency', ascending=False)

print("Mots les plus fréquents pour la classe 0 :")
print(class_0_freq)
```

Annexe 3 : Exemple de commentaire toxique

Annexe 4 : Exemple de commentaire non toxique