Bétry Steeven

Dandois Louis

# Natural Language Processing

## Objectif du Projet

Le projet vise à développer une solution automatisée pour l'analyse et la classification des commentaires postés par les utilisateurs sur une plateforme. L'objectif est de parvenir à entraîner un modèle de machine Learning capable de comprendre le contenu d'un commentaire et de le catégoriser en fonction de sa nature, qu'elle soit positive, neutre, ou négative, et de détecter d'éventuels contenus inappropriés.

## Démarche à Suivre

Pour atteindre cet objectif, plusieurs étapes sont préconisées :

Initialisation du projet : Création d'un répertoire GitHub pour le suivi et la sauvegarde du travail effectué.

Préparation des données : Utilisation d'un jeu de données spécifique pour l'entraînement du modèle, avec une approche initiale simplifiée pour évaluer la complexité du problème.

Développement itératif : Progression depuis un modèle de base vers des solutions plus avancées impliquant l'utilisation de techniques de Deep Learning, telles que l'embedding et les réseaux de neurones récurrents (RNNs).

Création d'un pipeline de traitement : Développement d'un système automatisé capable de prendre en charge l'analyse et la classification des commentaires depuis leur forme brute jusqu'à leur catégorisation.

## Les données et l'approche utilisées

Tout d'abord nous avons récupérer les données qui viennent du site wikipédia et qui ont été labellisées par le site kaggle en fonction de la toxicité des différents commentaires, ils ont été triés par différents labels qui sont les suivants : -toxic', 'severe_toxic','obscene','threat', 'insult', 'identity_hate'.

Nous avons seulement utilisé les données train afin de faire notre classification de ces commentaires. Grâce à cela nous avons pu commencer notre étude du jeu de données.

Pour cela, à la base il est intéressant de voir le nombre de commentaires négatifs en fonction de leurs catégories et les différentes statistiques que l'on peut avoir.

Ce qui peut être intéressant par la suite, c'est de voir graphiquement ce que chaque catégorie représente afin de pouvoir se donner une idée de la catégorie la plus représenté mais aussi celle qui comporte le moins de mots.

C’est donc pour cela que nous avons affiché un graphique représentant le nombre de mots par catégorie, nous pouvons remarquer ici que la catégorie ‘toxic’ est de loin la plus représenté.

Après avoir vu le nombre de mots par catégorie, nous voulions voir quels étaient les mots qui était le plus utilisés pour chaque catégorie. Pour cela nous avons utilisé des générations de nuages de mots.  Cela permet de savoir quels sont les mots les plus courant et cela nous permettra pour les tests à la fin du projet de savoir quels mots peuvent être utilisés pour avoir des tests concrets.

Et finalement, pour la dernière partie graphique, nous avons voulu voir le nombre de commentaires positif par rapport aux commentaires négatifs. C’est à ce moment-là que nous pouvons remarquer que les commentaires négatifs sont vraiment minimes et représente seulement une petite partie de tous les commentaires.

Après avoir représenté les données, il est nécessaire de préparer les données pour les utiliser ensuite avec nos modèles.

Nous avons trouvé un dictionnaire en regex qui va nous permettre de nous faciliter la tâche par la suite. Cela nous aidera à nettoyer les données en supprimant les nombres, les espaces, tous les caractères non-ascii etc.

Par la suite, pour continuer à nettoyer les données, nous avons utilisé la lemmatization et nous avons enlevé les stopwords.

La lemmatisation est le processus de réduction d'un mot à sa forme de base ou le lemme. Cela aide à regrouper différentes formes du même mot afin qu'elles puissent être analysées comme un seul élément, améliorant ainsi la précision des analyses de texte.

Les stopwords sont généralement des mots très communs dans une langue, comme "the", "is", "in". L'élimination des stopwords peut réduire le bruit dans les données et accélérer les traitements, permettant de se concentrer sur les mots qui portent réellement une information importante pour l'analyse.

Maintenant, nous devons faire le tokenizer pour transformer les mots en vecteur.

On va par la suite réduire le nombre de mots maximal ce qui est nécessaire pour certains modèles.

Avant de commencer l’entraînement des données, nous allons réduire le nombre de données que l’on va utiliser pour entrainer les modèles afin de réduire le temps de l’entraînement et éviter des temps d’attente qui peuvent durer des dizaines de minutes voire heures.

Nous avons d’abord commencé par un modèle bidirectionnel LSTM puis un modèle CNN et nous avons fini par le modèle GRU. Pour chaque modèle nous avons mis les graphiques de Loss et accuracy pour pouvoir les comparer et ainsi voir leur fiabilité et la précisions des modèles au cours de l’entraînement.

Finalement, nous mettons en œuvre les modèles avec différents commentaires pour voir quels sont les pourcentages que l'on obtient pour chaque catégorie.

## Les détails de notre solution

Notre objectif était de construire un système capable de détecter différents types de toxicité dans les commentaires en ligne, tels que la toxicité, la toxicité sévère, l'obscénité, la menace, l'insulte, et la haine identitaire. Pour ce faire, nous avons développé et comparé trois modèles distincts : Binary LSTM, CNN et GRU.

*Modèle Binary LSTM*

Le premier modèle que nous avons implémenté est basé sur les Long Short-Term Memory (LSTM) networks, une forme spécifique de réseaux de neurones récurrents (RNN) capable de capturer les dépendances à long terme dans les séquences de texte. Notre architecture Binary LSTM a été conçue pour traiter le texte séquentiellement, permettant au modèle de comprendre le contexte et la syntaxe sur de longues distances. Cette caractéristique est particulièrement utile pour la classification de commentaires, où le sens peut dépendre de l'ordre et de la relation entre les mots.

*Modèle CNN*

Le deuxième modèle exploite les réseaux de neurones convolutifs (CNN), plus couramment utilisés dans le traitement d'images, mais qui se sont également révélés efficaces pour le traitement du texte. Les couches convolutives permettent de détecter des motifs locaux dans les données, comme des phrases ou des expressions spécifiques pouvant indiquer la toxicité. Grâce à leur capacité à capturer ces caractéristiques spatiales locales, les CNN peuvent identifier des motifs de toxicité dans les commentaires sans nécessiter de comprendre le texte dans son intégralité.

*Modèle GRU*

Le troisième modèle utilise les Gated Recurrent Units (GRU), une variante des RNN similaire aux LSTM mais avec une structure plus simple. Les GRU offrent un équilibre entre la capacité de capture des dépendances temporelles et la complexité computationnelle, ce qui rend ce modèle à la fois performant et efficace pour l'analyse de texte. L'introduction de couches Dropout dans l'architecture GRU a été une stratégie clé pour réduire le surajustement, en permettant au modèle d'apprendre des représentations plus généralisables des données.

## Les améliorations possibles

Il y a beaucoup de pistes d'amélioration.

On aurait pu utiliser d'autres modèles comme BERT pour qu'il y ait une meilleure compréhension du texte. En effet, dans ce type de classification le contexte de certains mots dans la phrase est très important.

Les données n'ont pas l'air d'être assez diversifié pour nos modèles. Peut-être qu'un meilleur mélange ou en rassemblant + de mots sous un même synonyme (un peu comme le dictionnaire l'avait si bien fait) pourrait permettre une meilleure précision pour les modèles.

On n'a pas utilisé de modèles pré-entraînés (BERT encore par exemple, ou GPT), ce qui fait qu'il y a sûrement d'autres solutions au problème que l'on n'a pas exploré.

Un autre axe d'amélioration potentiel est la présence de majuscule. Certaines personnes les utilisent parfois pour crier mais par texte. C'est potentiellement intéressant d'y jeter un œil.

On avait aussi commencé à utiliser F1Score qui avait l'air pertinent comme indicateur pour ce type de problème, mais on a décidé finalement de rester sur la précision et d'abandonner l'idée. Peut-être que ce serait mieux de l'utiliser à l'avenir car on avait compris et ça avait l'air d'être vraiment un bon moyen de voir si notre modèle est efficace.

Notre modèle GRU est peu efficace. Il sait faire la différence entre toxique ou non, mais pas selon les différents niveaux dans la toxicité elle-même. De plus, il avait overfit au départ ce qui indique une mauvaise gestion par rapport à celui-ci qui est donc moins efficace que les autres.