NATURAL LANGUAGE PROCESSING

PROJET : CLASSIFICATION DE COMMENTAIRES

PROFESSEUR : *M. MORGAN GAUTHEROT*

ETUDIANTS *: PAUL MAZAINGUE*

MARLENE MEVAH

1. Contexte et sujet du projet

Aujourd'hui, les discussions en ligne font partie intégrante de notre quotidien, mais elles peuvent parfois devenir toxiques. De nombreuses plateformes doivent gérer des millions de commentaires et identifier ceux qui contiennent des propos haineux ou inappropriés.

C'est là que notre projet intervient ! L'objectif est de développer un modèle d'intelligence artificielle capable de classifier automatiquement les commentaires en différentes catégories de toxicité. Ce modèle facilitera le travail des modérateurs et contribuera à rendre les espaces de discussion plus sûrs pour tous.

2. Données et approche utilisées

2.1 Jeu de données

Nous avons utilisé les données du Comment **Classification Challenge**, disponibles sur Kaggle. Ce jeu de données contient une grande variété de commentaires annotés en fonction de leur niveau de toxicité (toxique, sévèrement toxique, insultant, haineux, etc.).

Dès le début, nous avons remarqué que la majorité des commentaires étaient non toxiques, ce qui nous a poussé à adapter notre approche pour bien équilibrer l'apprentissage du modèle.

2.2 Approche méthodologique

Notre démarche a été progressive, en suivant ces étapes :

1. **Prétraitement des données** : Nettoyage du texte pour supprimer les caractères spéciaux, mettre tout en minuscules et éliminer les mots inutiles (stopwords).
2. **Exploration des données** : Visualisation et analyse des distributions pour mieux comprendre la répartition des classes.
3. **Modélisation** :

    a. D'abord, une première approche simple en utilisant un modèle basé sur un LLM pour obtenir rapidement des résultats.

b. Ensuite, un modèle plus avancé utilisant **des embeddings et des réseaux de neurones récurrents (RNNs)** pour mieux capter le sens des phrases.
c. Enfin, nous avons comparé ces deux approches indépendamment :
    i. Le **LLM** a été utilisé comme une solution rapide et prête à l'emploi pour classifier les commentaires en interrogeant une API.
    ii. Le **RNN** a été entraîné à partir des embeddings pour capturer des relations plus complexes dans les textes.
    iii. Nous avons observé que le LLM donne des résultats rapides mais peut être limité dans sa capacité à bien généraliser sur nos données spécifiques, tandis que le RNN demande plus d'entraînement mais pourrait potentiellement mieux s'adapter si les données sont bien équilibrées et nettoyées.

4. **Évaluation** : Grâce aux métriques AUC-ROC et F1-score, nous avons pu comparer les performances des différentes approches.
5. **Mise en place d'un pipeline** : Un système complet permettant de soumettre un commentaire et d'obtenir une classification immédiate.

3. Détails de la solution

3.1 Prétraitement des données

Un bon modèle commence par de bonnes données ! Nous avons donc :

- Converti tous les textes en minuscules.
- Supprimé les caractères spéciaux et les mots inutiles.
- Utilisé des embeddings de mots (Word2Vec, GloVe) pour représenter les commentaires de manière plus efficace.

3.2 Modèles utilisés

Nous avons testé plusieurs approches :

- **Baseline** : Une première évaluation avec un modèle basé sur un LLM, permettant de classifier directement la toxicité des commentaires.
- **Modèle avancé** : Un **Réseau de Neurones Récurrent (RNN)**, utilisant des embeddings, capable de mieux comprendre le contexte des phrases.
- **Approche LLM** :
    - Nous avons exploité l'API Google Generative AI avec le modèle **Gemini 1.5 Flash**.

- Une fonction `is_toxic(text)` envoie un prompt à l'API pour demander si un commentaire est toxique ou non.
- L'API retourne `1` si le commentaire est toxique, `0` sinon.
- Cette solution est efficace mais dépend d'une API externe.

3.3 Pipeline d'inférence

À la fin, nous avons construit un pipeline qui permet de :

- Prendre une phrase en entrée.
- Effectuer tous les traitements nécessaires.
- Donner une classification en sortie.

4. Améliorations possibles

Même si notre modèle donne déjà de bons résultats, voici quelques pistes pour aller encore plus loin :

- **Utiliser BERT ou d'autres modèles NLP avancés** pour améliorer la compréhension du contexte.
- **Augmenter les données** en appliquant des techniques de data augmentation pour équilibrer les classes.
- **Optimiser les hyperparamètres** avec des techniques comme GridSearch pour affiner les performances.
- **Déployer le modèle** sous forme d'API ou d'application web pour qu'il puisse être utilisé en temps réel.
- **Réduire la dépendance à l'API externe** en entraînant un modèle local aussi performant qu'un LLM.

Conclusion

Ce projet a été une belle opportunité d'explorer les défis de la classification de texte et de mettre en œuvre différentes approches pour améliorer la détection de toxicité dans les commentaires en ligne.