Note méthodologique : preuve de concept

Dataset retenu

Le dataset utilisé pour ce projet comprend 50 000 questions extraites du site Stack Overflow via l'outil StackEchange Data Explore. Stack Overflow est une plateforme populaire pour les développeurs où ils peuvent poser des questions et obtenir des réponses sur divers sujets de programmation. Le dataset est varié et couvre une multitude de langages de programmation, de frameworks et de problématiques techniques, ce qui le rend idéal pour des tâches de classification de texte et d'analyse du langage naturel. Les questions sont généralement bien structurées, incluant des titres (Title), des descriptions détaillées (Body), et parfois des étiquettes (Tags) qui indiquent les sujets abordés.

Le dataset contient au total 8 features dont 1 feature float64, 4 features int64 et 3 features object. Mais tout au long de notre travail, seuls les feature Title, Body et Tags seront utilisés. Afin d'assurer une qualité des données, les contraintes suivantes ont été établie lors de l'extraction :

- *Minimum une réponse*
- *Un score supérieur à 5*
- *Au moins 20 vues*

Dans façon détaillée, les features présentes dans notre dataset sont :

- Title : Le titre de la question.
- Body : Le contenu complet de la question.
- Tags : Liste des tags associés à la question.
- Id : identité de la question
- Score : score obtenu par la question auprès des visiteurs
- ViewCount : nombre de vues obtenues par la question
- FavoriteCount **:** nombre de fois que la question a été placée comme favorite

- ❖ Accepted Answer : La réponse acceptée (le cas échéant)

Une fois recueillies, un prétraitement a été appliqué aux questions en vue de le rendre prêtes à la modélisation. Comme prétraitement, on peut citer entre autres : le nettoyage du texte (Suppression des balises HTML, ponctuation, et autres caractères spéciaux), la tokenisation (séparation du texte en mots ou tokens) et l'encodage (la transformation des tokens en vecteurs numériques utilisables par les modèles de machine learning)

Les concepts de l'algorithme récent

SciBERT-CNN :

SciBERT (Scientific BERT) est une variante de BERT pré-entraînée sur des corpus scientifiques. Il est particulièrement efficace pour les tâches de traitement de langage naturel dans des domaines techniques ou scientifiques.

Principes de Fonctionnement :

SciBERT : Utilise un modèle transformeur pour capturer les relations contextuelles entre les mots dans une phrase. SciBERT est pré-entraîné sur des textes scientifiques, ce qui le rend plus performant pour les données techniques.

CNN (Convolutional Neural Network) : Après l'encodage des tokens par SciBERT, un CNN est appliqué pour capturer les caractéristiques locales et les motifs dans les séquences de tokens. Les CNN sont efficaces pour détecter les relations locales entre les mots ou les phrases.

Architecture :

Input Layer : Prend les tokens encodés par SciBERT.

Convolutional Layer : Applique plusieurs filtres pour extraire les caractéristiques locales.

Pooling Layer : Réduit la dimensionnalité et maintient les caractéristiques importantes.

Fully Connected Layer : Combine les caractéristiques extraites pour prédire les tags.
Output Layer : Utilise une fonction softmax pour fournir les probabilités des tags possibles.

Avantages :
Contextual Understanding : SciBERT capture les relations contextuelles entre les mots.
Feature Extraction : CNN extrait efficacement les caractéristiques locales pertinentes.
Adaptabilité : Performant pour les données textuelles techniques et scientifiques.

La modélisation

Méthodologie de Modélisation :
Prétraitement des Données : Nettoyage, tokenisation, et encodage des questions.
Entraînement du Modèle : Utilisation de SciBERT pour encoder les tokens, suivi par l'application d'un CNN pour extraire les caractéristiques et prédire les tags.
Validation Croisée : Utilisation de la validation croisée pour évaluer la performance du modèle et éviter le surapprentissage.

Métrique d'Évaluation :
Accuracy : Proportion des prédictions correctes.
F1-Score : Moyenne harmonique de la précision et du rappel, particulièrement utile pour les classes déséquilibrées.

Démarche d'Optimisation :
Ajustement des Hyperparamètres : Ajustement des hyperparamètres comme le taux d'apprentissage, la taille des lots (batch size), et le nombre de filtres CNN.

Techniques de Régularisation : Utilisation de dropout et de normalisation pour éviter le surapprentissage.
Fine-tuning : Ajustement des poids du modèle SciBERT pré-entraîné pour mieux s'adapter aux spécificités du dataset de Stack Overflow.

Une synthèse des résultats

Comparaison des Performances :

Modèle	Precision	F1-Score	Accuracy	Recall	Training_time
Logistic Classifier	71%	66%	**55%**	**63%**	**269.08 s**
SciBERT-CNN	82%	**76%**	**71%**	**70%**	**5h30mn**

Conclusion :

Les résultats montrent une amélioration significative des performances avec l'utilisation de SciBERT-CNN par rapport au logistic classifier. La precision a augmenté de 11%, le F1-Score de 11%, l'accuracy de 16% et le ReCall de 7%, indiquant une meilleure gestion des classes déséquilibrées.

L'analyse de la feature importance globale et locale du nouveau modèle

Feature Importance Globale :

Analyse des Poids : Identification des poids des filtres CNN les plus importants pour la classification.
Importance des Tokens : Utilisation de techniques comme LIME ou SHAP pour déterminer les tokens les plus influents dans les décisions du modèle.

Feature Importance Locale :

Exemples Concrets : Analyse de cas spécifiques où le modèle SciBERT-CNN a fait des prédictions correctes ou incorrectes, en identifiant les caractéristiques textuelles qui ont influencé ces décisions.

Visualisation : Utilisation d'outils de visualisation pour montrer l'importance des tokens dans les prédictions locales.

Les limites et les améliorations possibles

Limites :

Complexité Computationnelle : SciBERT-CNN est plus exigeant en termes de ressources computationnelles que les modèles plus simples.

Données Imparfaites : Les erreurs dans les données d'entrée (questions mal formulées, tags incorrects) peuvent affecter les performances du modèle.

Améliorations Envisageables :

Entraînement sur des Données Plus Grandes : Utilisation d'un dataset plus large pour entraîner le modèle.

Affinage du Modèle : Exploration de variantes de BERT ou d'autres architectures de réseaux neuronaux pour encore améliorer les performances.