IFT 714 - Traitement automatique des langues naturelles
Groupe 1 : Détection des fake news par fact checking
Rapport d'avancement

Anne-Sophia Lim, Yanis Perrin, Caroline Wang

1 Introduction

L'avènement d'Internet et des médias sociaux a bouleversé notre manière de consommer l'information, facilitant la propagation mondiale d'informations non vérifiées. La désinformation, notamment autour de sujets comme la COVID-19, met en évidence l'importance de vérifier les informations diffusées. Pour cela, les plateformes mettent en place des méthodes de détection de fausses informations telles que l'analyse textuelle et les algorithmes de traitement automatique du langage. Cependant, ces méthodes ont des limites en raison de l'évolution constante du langage et du défi de créer des bases de données exhaustives sur les fausses informations. Ce projet vise à implémenter un modèle de classification permettant de déterminer si les informations véhiculées sont factices ou non et de mesurer l'impact d'une augmentation de données sur les résultats obtenus.

2 Etat de l'art

L'un des plus gros défis du traitement de texte réside dans le traitement et l'extraction des caractéristiques qui seront données au modèle. De nombreuses études se basent principalement sur la représentation du texte via des méthodes comme TF-IDF ou des modèles N-gram (Ahmed et al., 2017). Cependant dans ce projet, il pourrait aussi être intéressant d'extraire des caractéristiques lexicales et sentimentales qui pourraient aider le modèle à mieux comprendre le texte.

L'augmentation de données textuelles est une technique d'apprentissage automatique pour enrichir un ensemble de données existant en créant de nouvelles données à partir des données d'origine. L'objectif principal est d'améliorer la capacité de généralisation d'un modèle en l'exposant à une plus grande variété de données pendant l'entraînement. Il existes différentes méthodes.

L'EDA (Easy Data Augmentation) repose sur quatre transformations de base pour chaque phrase (Wei and Zou, 2019). La **substitution de synonymes** aléatoire de n mots de la phrase par l'un de ses synonymes. L'**insertion aléatoire** d'un synonyme de n mots. La **permutation aléatoire** de deux mots dans la phrase, effectuée n fois. La **suppression aléatoire** de chaque mot de la phrase avec une probabilité p. Les limites de l'EDA peuvent se faire ressentir dans le sens où elle implique généralement un ensemble limité de transformations. Il est également possible d'utiliser des modèles transformers préentrainés (Kumar et al., 2020) tels que BERT ou BART, leur avantage étant qu'ils ont été testé sur d'autres bases de données. Cependant, ces modèles requiert néanmoins un volume relativement important de données pour bien généraliser, nous procéderons donc à une augmentation des données à l'aide de la méthode EDA pour entraîner nos modèles.

3 Méthodologie

Base de données

La base de données choisie est la base LIAR (Liar Dataset) (Wang, 2017). Elle contient plus de 12 000 citations politiques factuelles provenant du site internet POLITICFACT.COM[1] avec des métadonnées telles que le sujet, le contexte, l'auteur mais aussi l'historique de la véracité des faits qu'il a énoncés précédemment. Chaque déclaration est étiquetée par un éditeur du site avec l'une des six valeurs possibles de véracité, allant de "pants-fire" (mensonge éhonté) à "true" (vrai). Dans le présent projet, l'historique de véracité contenant la majorité des informations liés à l'auteur les informations suivantes sont conservées : l'étiquette, la citation et l'historique de véracité.

Prétraitement des données

Avant d'entamer la classification, il est essentiel d'effectuer une étape de prétraitement afin de ré-

[1] https://www.politifact.com/

duire la taille de nos données pour pouvoir en tirer les informations les plus pertinentes pour notre tâche de classification. Cette étape de prétraitement contient notamment la tokenisation ainsi que la lemmatisation du texte mais aussi la gestion des mots non significatifs.

La **tokenisation** du texte consiste à découper le texte en unités plus petites, appelées tokens, qui peuvent ensuite être traitées par nos modèles. Pour ce faire, nous utilisons la bibliothèque python bien établie Natural Language Toolkit (nltk).

Une fois les tokens récupérés, il est important de supprimer les **mots non significatifs**, qui n'apportent pas ou très peu d'informations comme *a, about, by, for, from etc*. Cette suppression est dûe au fait que les mots non significatifs pourraient apporter du bruit non désiré lors de la classification du texte.

La **lemmatisation** cherche à transformer les tokens jusqu'à leurs formes fondamentales, cette forme résultante est appelée « lemma ». Par exemple, lors de la lemmatisation, les tokens "runs", "running" et "ran" sont réduits au lemma "run". Cela nous permet de décroitre le nombre de types de mots dans nos données et d'obtenir une classification plus rapide et efficace.

Nous avons extrait plusieurs **caractéristiques lexicales** comme le nombre de caractères liés à la ponctuation mais aussi le nombre de mots et de digits dans les citations. Il est aussi intéressant d'extraire le nombre d'étiquetage des mots dans un texte avec leurs parties du discours correspondantes (verbes, adjectif, pronom. . . etc). Cela pourrait aider les algorithmes à comprendre la structure grammaticale et la signification de nos citations. Enfin, nous avons extrait une **analyse sentimentale** en évaluant les sentiments dégagés par la citation (positif, neutre et négatif).

Les algorithmes modernes de traitement automatique des langues utilisent les "embeddings" comme représentation du sens des mots. Ce sont des vecteurs sémantiques qui définissent le sens d'un mot comme un point dans un espace de dimension finie. Dans notre projet, nous utilisons la représentation **TF-IDF** afin de décomposer les citations en vecteurs éparses traitable par le modèle de classification. Par définition, les mots sont définis par leurs environnements (les mots qui les entourent). TF-IDF utilise cette définition en représentant les mots par une pondération du décompte des mots voisins.

Modèle de référence et Méthodes d'évaluation

Notre modèle de référence est une machine à vecteur de support linéaire (LSVM), d'après certaines recherches (Ahmed et al., 2017), celui-ci semble très bien performer sur le LIAR dataset. Afin d'évaluer les résultats, nous récupérons trois fichiers csv déjà existants (train, valid, test) qui servent respectivement à l'entraînement, la validation ainsi que le test de nos modèles. En ce qui concerne les métriques d'évaluation, afin de ne pas se limiter à la simple exactitude du modèle nous utilisons aussi le rappel, la précision et le score F1.

4 Résultats

L'EDA, en créant une nouvelle observation à partir de chaque donnée, nous a permis de doubler nos données. Les résultats ci-contre sont obtenus à partir d'une tâche de classification binaire.

Table 1: Résultat après entrainement des modèles

Modèles	Sans EDA				Avec EDA			
	A.	P.	R.	F1	A.	P.	R.	F1
LSVM	0.70	0.72	0.93	0.81	0.83	0.83	0.94	0.88
LR	0.72	0.72	0.98	0.81	0.73	0.73	0.98	0.84
RF	0.73	0.73	0.99	0.74	0.90	0.87	0.99	0.93

On peut observer que le modèle le plus performant est le random forest. L'augmentation des données a permis l'améliorations de toutes les métriques d'évaluations, ce qui semble être une grande réussite. Cependant, il se pourrait qu'en créant de nouvelles observations trop similaires aux données initiales les résultats se retrouvent faussés car si le modèle prédit correctement pour une données alors il prédira aussi correctement pour la donnée crée.

5 Travaux futurs

L'objectif serait ensuite de tester une autre manière d'augmenter les données, par exemple en créant des paraphrases à l'aide de modèles préentrainés permettant de paraphraser des phrases données en entrée (Sutskever et al., 2014). Cela permettrait d'avoir des phrases assez différentes après l'augmentation pour ne pas fournir des données trop similaires les unes des autres. D'autres pistes sont envisagées afin d'améliorer la précision de nos modèles comme l'utilisation de la validation croisée ou de l'algorithme grid search afin de trouver les meilleurs hyperparamètres des modèles. Enfin, il serait peut être intéressant de tester d'autres types d'embeddings comme la représentation Word2Vec (Mikolov et al., 2013).

References

Hadeer Ahmed, Issa Traore, and Sherif Saad. 2017. Detection of online fake news using n-gram analysis and machine learning techniques. pages 127–138.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 18–26, Suzhou, China. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks.

William Yang Wang. 2017. "liar, liar pants on fire": A new benchmark dataset for fake news detection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 422–426, Vancouver, Canada. Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Contribution des membres

Anne-Sophia : Recherche de propositions. Recherche sur les enjeux du sujet. Explication des problématiques posées par le sujet. Introduction et état de l'art. Description des données.

Yanis : Recherche de propositions. Recherche sur les bases de données disponibles. Explication du contexte et de l'importance du projet. Prétraitement de la base de données, extraction des caractéristiques et implémentation des modèles.

Caroline : Recherche de propositions. Recherche sur les modèles applicables pour le projet. Explication de la méthode adoptée. Description des données. Implémentation de l'augmentation de données.

Ressources Informatiques

Dans notre projet, python est le langage de programmation utilisé, celui-ci est le plus utilisé et conseillé dans la manipulation des données. Nous faisons appel à plusieurs librairies python dans ce projet (Numpy, Pandas, PyTorch, Scikit-Learn, NLTK). Numpy et Pandas nous facilitent la manipulation des données grâce aux diverses méthodes qu'elles présentent. NLTK est une libraire utilisée pour le traitement du langage naturel. PyTorch et Scikit-Learn nous permettront de créer, d'entraîner et de tester nos modèles. Enfin, nous utilisons Jupyter Notebook qui est une plate-forme informatique interactive basée sur le Web qui nous permet de tester notre code via des cellules de codes. Github est utilisé pour le partage et la gestion des versions du projet. Le jeu de données étant de taille raisonnable, nous devrions pouvoir faire tourner notre projet sans problème sur nos ordinateurs personnels.