REPUBLIQUE TUNISIENNE
Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique

ÉCOLE SUPERIEURE PRIVÉE
DE TECHNOLOGIE & INGENIERIE

Cycle de Formation des Ingénieurs en Informatique

Rapport de Projet

Détection des mots clés et classification des articles COVID

Elaboré Par

Elahmer Intissar

Maalaoui Maha

En collaboration avec

Année Universitaire : 2021/2022

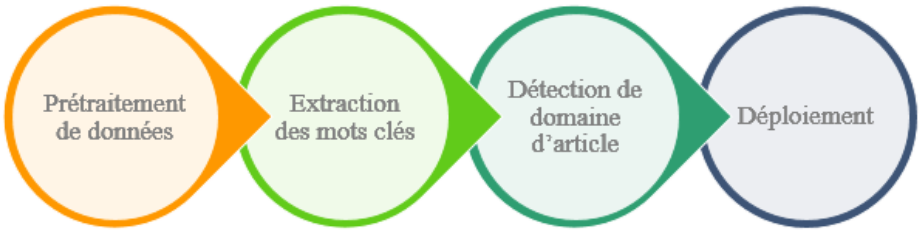
Introduction :

Dans les articles, les mots-clés constituent un élément important puisqu'ils représentent le contenu de l'article. Les mots clés jouent également un rôle crucial dans la localisation de l'article à partir des systèmes de recherche d'informations, des bases de données bibliographiques et pour l'optimisation des moteurs de recherche. Les mots-clés permettent aussi de classer l'article en fonction du sujet ou de la discipline.

Dans ce cadre notre projet consiste à appliquer les techniques de NLP (Traitement du langage naturel) et de Deep learning pour aider à extrait les mots clés d'un article et déduire si l'article porte sur covid-19 ou non.

Puisque l'objectif de notre projet est la compréhension du concept d'extraction de mots clés, l'utilisation du texte complet de l'article conduit une grande complexité de calcul, on va utiliser seulement les résumés pour la modélisation NLP. Le même travail peut être appliquer sur tout le texte de l'article pour obtenir une meilleure extraction de mot-clé.

Dans Notre Travail on va suivi le processus suivant



1. Prétraitement de données :

Cette partie a comme un objectif de nettoyer et normaliser les données pour avoir une matrice de vocabulaire efficace et représentatifs de nos documents.

Ce traitement inclus :

- Elimination des « stops words » avec les prédéfini English stop_words on a défini une liste personnaliser des mots qui sont fréquent dans les articles scientifiques.
- Supprimer les ponctuations.
- Convertir en minuscules.
- Supprimer les caractères et chiffres spéciaux.
- Lemmatisation: consiste à rendre n'importe quel type de mot vers son forme de base.

```
sph calculation mar scale collision role equation state material rheology numerical effect model scale approx km impact mar like planet smoothed
particle hydrodynamics code effect material strength different equation state post impact material temperature distribution investigated property
ejected material term escaping disc mass analysed study potential numerical effect context density discontinuity rigid body rotation scale collision
regime considered impact velocity km effect material strength substantial post impact distribution temperature impactor material influence equation
```

Figure 1 Texte après normalisation

2. Extraction des mots clés :

Pour l'extraction de mots clés, nous avons utilisé 3 méthodes : TF_IDF, Spacy, Yake.

a) Bag of words :TF_IDF

« Bag of words » pour extraction des features. Bag of words est une représentation de texte qui décrit l'occurrence de mots dans un document toute information sur l'ordre ou la structure des mots dans le document est rejetée. Le modèle ne porte que sur la présence de mots connus dans le document.

Le bag of words nécessite 2 choses:

- Un vocabulaire de mots connus.
- Mesure de la présence de mots connus.

❖ Création de vocabulaire :

Pour construire le vocabulaire de mots connus on a utilisé la fonction CountVectoriser et on a le paramétré pour abandonner les mots qu'ont une fréquence dans le document supérieur à 80%. Cela permet de s'assurer que nous n'avons que des mots pertinents au contexte. De plus on va exploite pas seulement des mots simples mais aussi combinaison de 2 mots (bi-grams) et 3 mots (tri-gram).

❖ Mesure de la présence de mots connus:

Cette étape est pour affiner le vocabulaire de mots en utilisant le TF-IDF vectoriser. La limite de vocabulaire obtenu du CountVectoriser est que, grands nombres de certains mots communs peuvent diluer l'impact de mots plus spécifiques au contexte dans le corpus. Ceci est corrigé par TF-IDF vectoriser qui sont des scores de fréquence de mots qui mettent en évidence les mots les plus importants pour le contexte que ceux qui apparaissent souvent dans les documents.

La TF-IDF comprend deux composantes :

- TF: Fréquence des termes
- IDF: Fréquence inverse des documents

$$\bullet\ \mathrm{TF} = \frac{\text{Frequency of a term in a document}}{\text{total number of terms in the document}}$$

$$\bullet\ \mathrm{IDF} = \frac{\log(\text{Total documents})}{\text{\# of documents with the term}}$$

En basent sur les scores TF-IDF, nous pouvons extraire les mots ayant les scores les plus élevés pour obtenir les mots clés d'un document. On a ordonner les scores TF-IDF en ordre descendent et sélection les 10 top mots comme keywords.

b) SpaCy :

SpaCy est une bibliothèque open-source pour le traitement du langage naturel (NLP) en Python avec beaucoup de capacités intégrées. Il est de plus en plus populaire pour le traitement et l'analyse des données dans le NLP. SpaCy offre des modèles avec des vocabulaire pré-trainé qui permettent l'extraction des mots comme « en_core_sci_sm » qu'on a utilisé dans notre projet.

c) Yake :

Yake (Yet Another Keyword Extractor) est une approche non supervisée pour l'extraction automatique de mots-clés à l'aide de Text Features. C'est une méthode légère d'extraction automatique de mots-clés qui repose sur des caractéristiques statistiques extraites de documents uniques pour sélectionner les mots-clés les plus importants d'un texte. Le système n'a pas besoin d'être formé sur un ensemble particulier de documents, ni il dépend des dictionnaires, du corpus externe, de la taille du texte, du langage ou du domaine.

3. Détection de domaine d'article :

Dans cette partie on va utiliser les techniques de Deep Learning pour classifier un article en COVID et Non COVID avec l'utilisation des mots clés.

a) Préparation des donnes :

Pour ce projet on a utilisé deux base de données à partir de Kaggle. Une base de données qui contient des articles de covides-19 et une autre qui contient des articles scientifique (computer science, physique, math ...). Chaque base de données contient 2 champs titre et abstract.

Après faire les vérifications nécessaires, on a trouvé que nos bases de données ne contiennent pas des valeurs manquantes ou redoublantes.

Puis on a concaténé pour chaque base les champs abstract et titre pour améliorer l'extraction des mots clés et on a aussi concaténer les deux bases, crée les labels et on a effectué un shuffle sur la nouvelle base.

Dans notre cas la classification est une classification binaire.

b) Exploitation des données :

Dans l'étape suivante, on a calculé le nombre de mots par chaque échantillon. Le nombre de mots est important pour nous donner une indication sur de la taille de données que nous allons traiter ainsi que de la variation du nombre de mots à travers les échantillons. On a trouvé que le nombre moyen de mots est 212 mots par abstract + titre. Le nombre de mots varie d'un minimum de 13 à un maximum de 568.

Dernièrement, on a trouvé les mots les plus fréquentes et les moins fréquentes. On a trouvé que les mots les plus fréquent sont les « English stop_words », aussi on a trouvé une liste de mots qui doivent être ajoutés aux liste de « stop_words », et pour les moins fréquentes sont des chaines de caractères spéciaux.

Après normalisation de nos données, on a fait une visualisation des mots les plus fréquentes, en créant ce qu'appel « Word cloud »

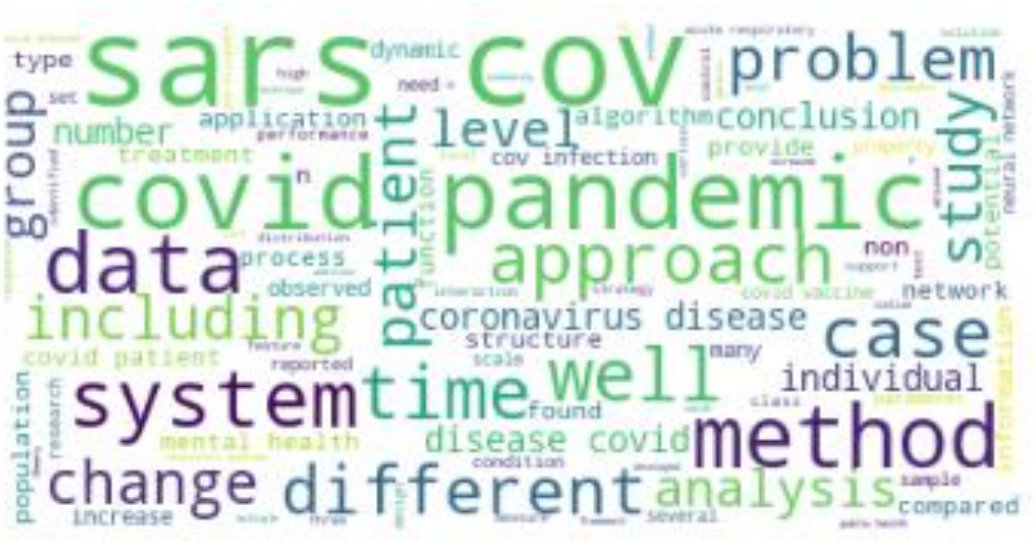


Figure 2 les mots les Plus fréquents

En utilisant le model Bag of words et count CountVectorizer pour exploiter et visualiser le vocabulaire de mots et les mots les plus fréquents présenter dans notre base et de plus on a exploité pas seulement des mots simples mais aussi combinaison de 2 mots (bi-grams) et 3 mots (tri-gram).

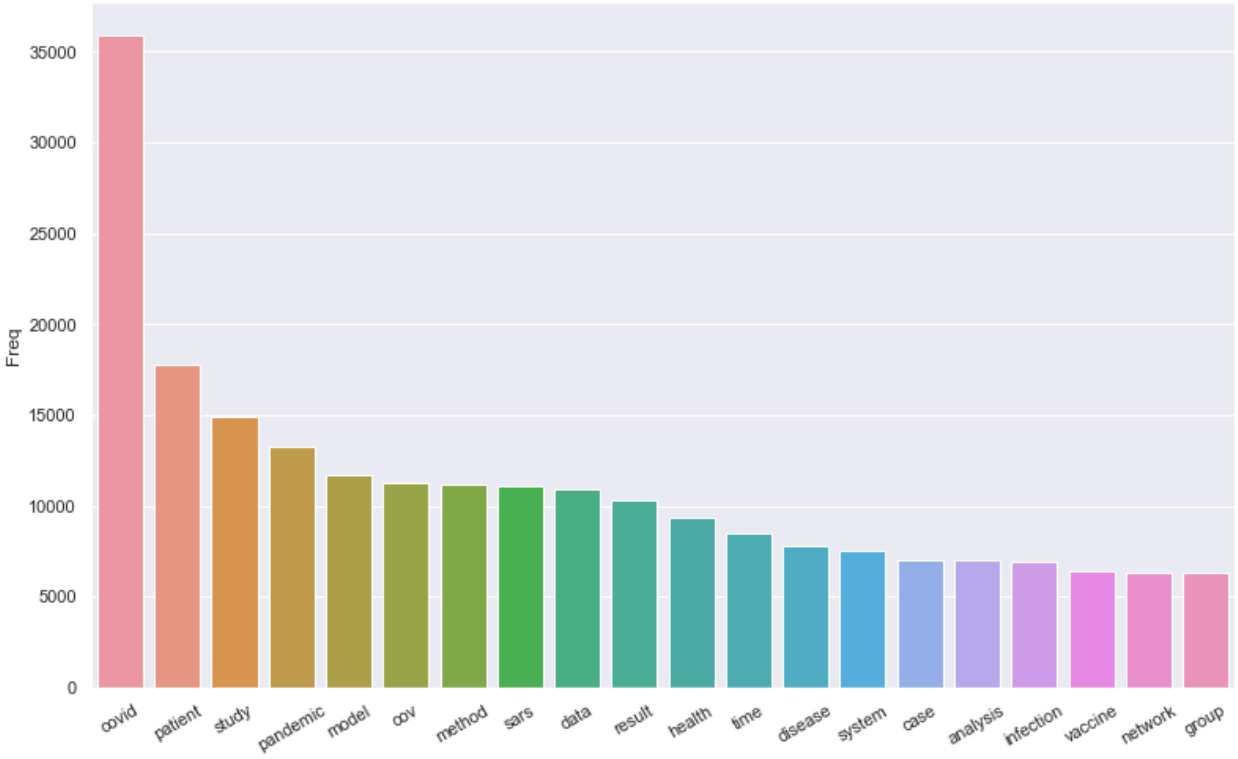


Figure 3 les uni-grams les plus fréquentes

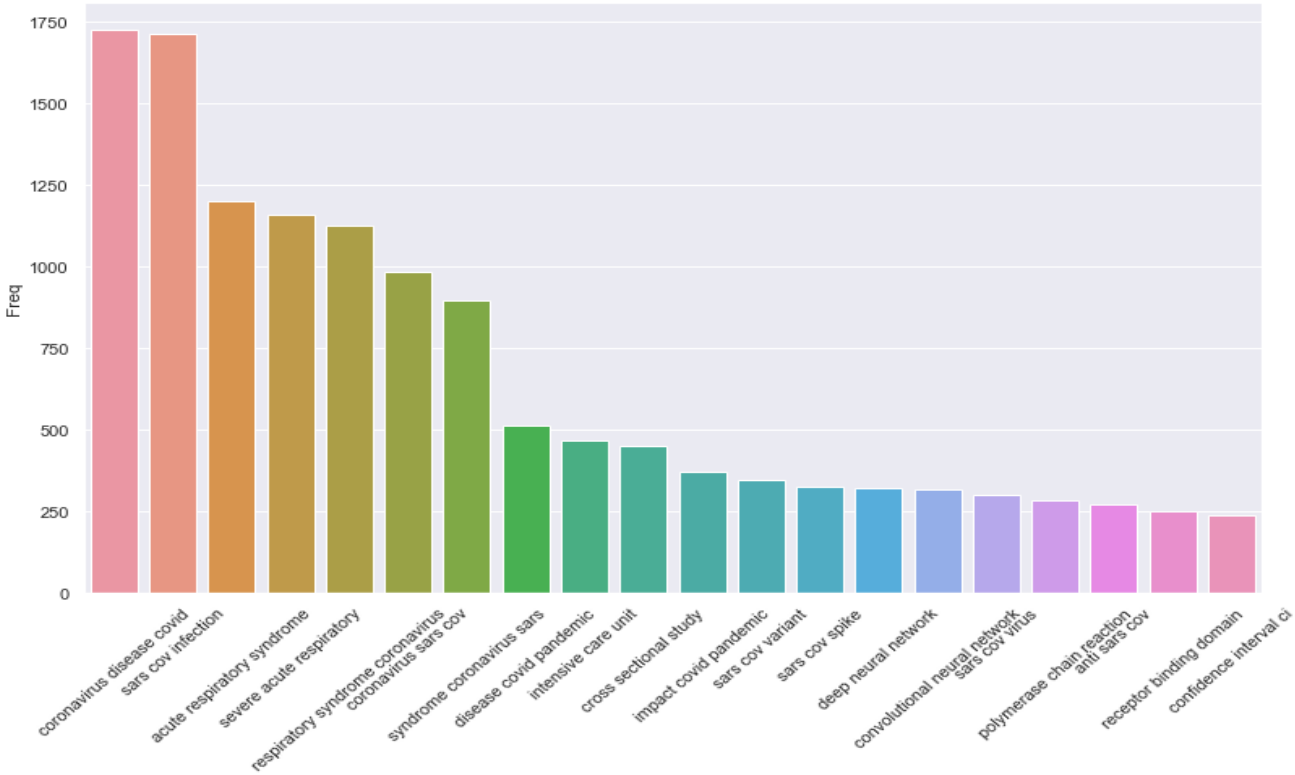


Figure 4 les tri-grams les plus fréquentes

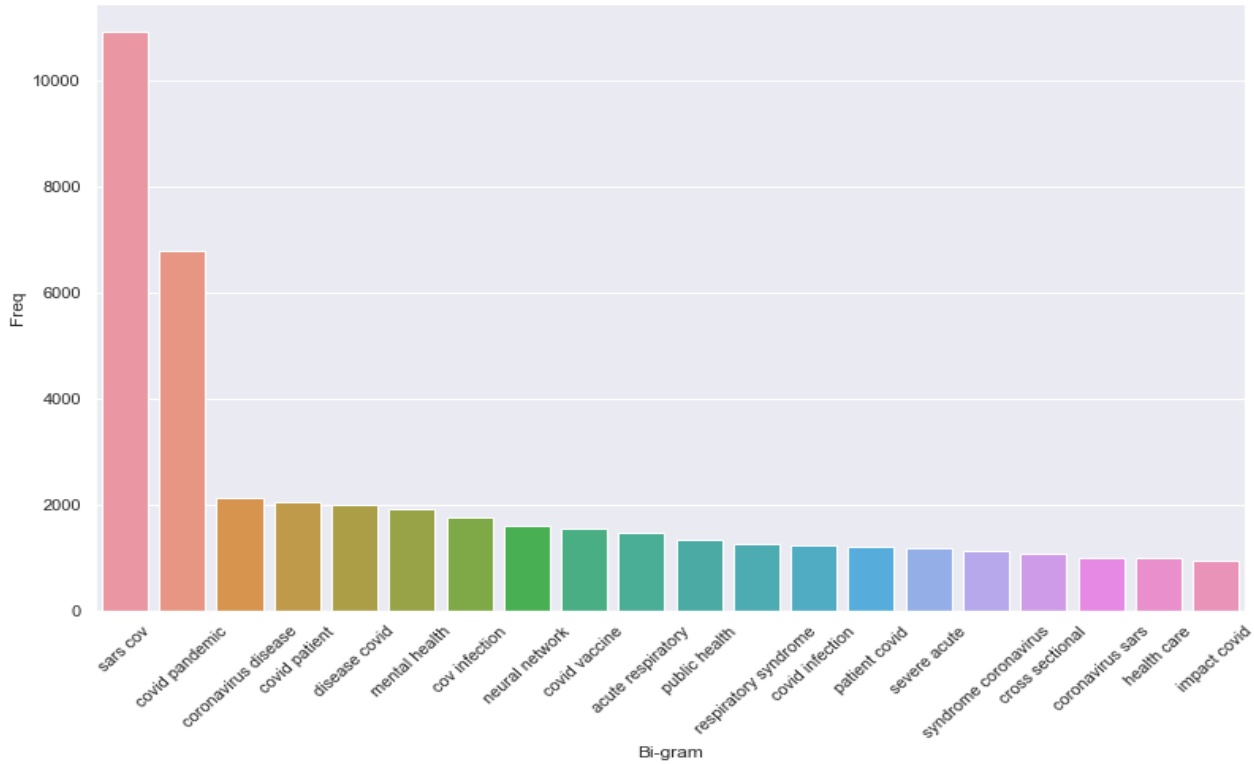


Figure 3 les bi-grams les plus fréquentes

Enfin on a utilisé la méthode « Tokenizer » pour préparer les données au modelés de Deep Learning.

c) Construit de model

Dans cette partie on a utilisé et testé les model LSTM, BLSTM, GRU de la class RNN (Recurrent Neural Network). L'architecture générale de nos model ce compose de :

- Un sequential Layer
- Un Embedding Layer
- Un LSTM Layer
- 2 éme dense

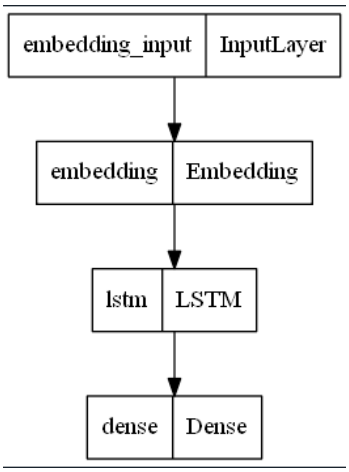


Figure 4 LSTM model

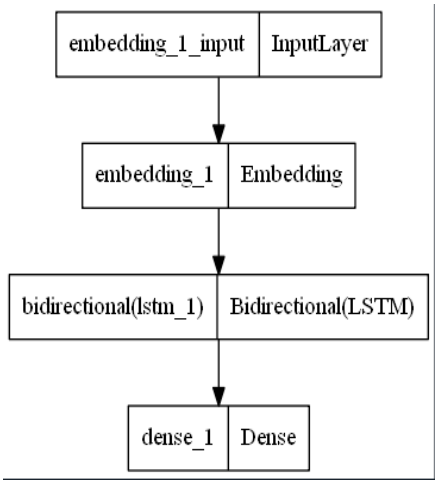


Figure 7 BLSTM model

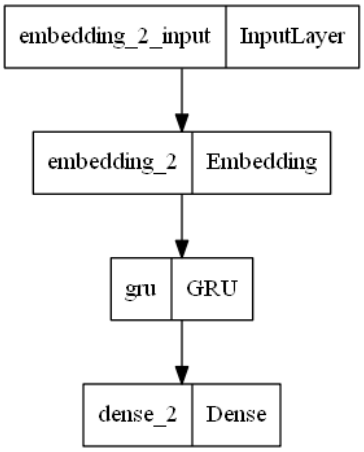


Figure 8 GRU model

d) Evaluation

Après la phase de train, on passe à l'évaluation de performance. Dans notre projet on a utilisé 3 modèles LSTM, BLSTM, GRU. On va comparer la performance de ces trios modèles selon accuarcy et choisi le model le plus performant pour le déploiement.

Le tableau suivante présente les différents valeurs d'accuarcy pour les 3 modèles. Selon ce tableau et la courbe de Roc (figure 5) on peut conclus que le modèle BLSTM est le plus performant et don on va le choisi pour le deploiement.

	LSTM	BLSTM	GRU
Accuarcy	0.94	0.98	0.95

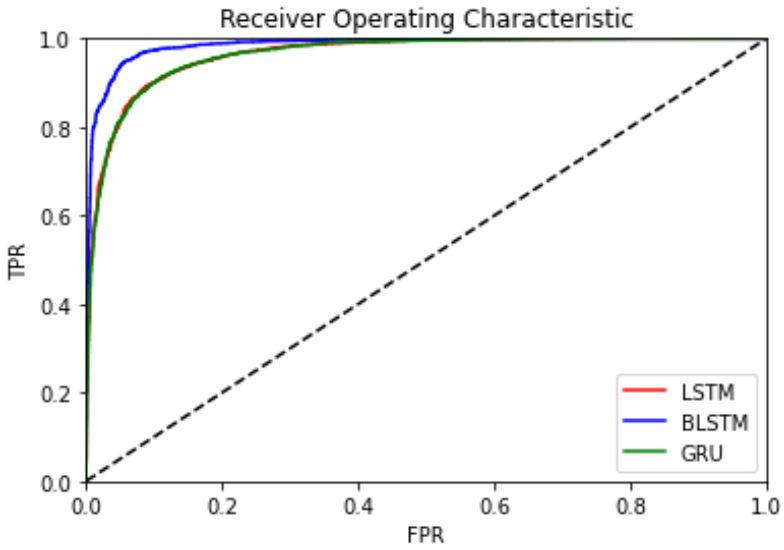


Figure 5 comparaison de LSTM, BLSTM et GRU

4. Déploiement :

Pour déploiement de notre application on utiliser Streamlit avec Visual studio.

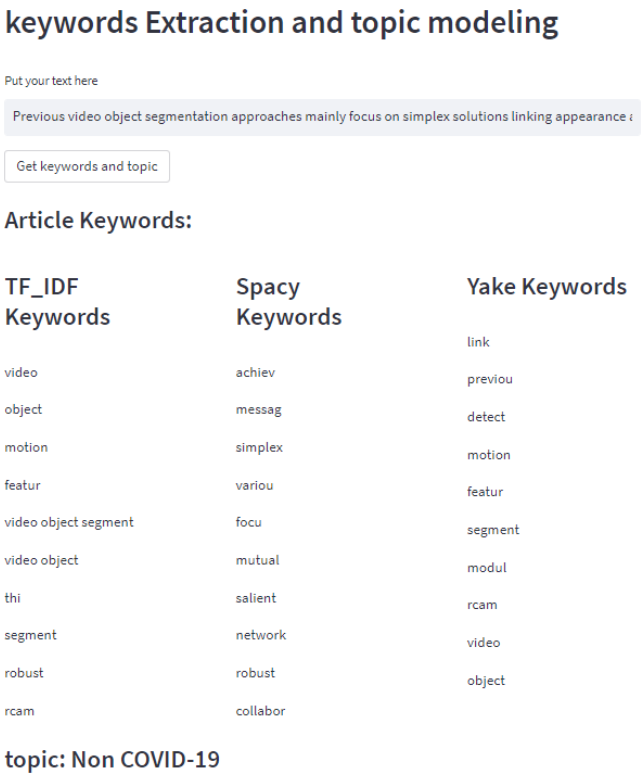


Figure 6 interface de l'application

5. Conclusion

Dans ce projet on a essayé de faire une extraction des mots clé des articles. On a utilisé le model bag of words et le technique TF-IDF et les techniques Spacy et Yak. Puis on a construit un modèle de classification pour dégage les articles de Covid-19 en utilisant et en tester les 3 méthode LSTM, BLSTM, GRU de RNN.