

**UNIVERSITÉ MOHAMMED V**  
**Faculté des Sciences de Rabat**



**Master Ingénierie de Données et  
Développement Logiciel**

*Natural language Processing and Applications*

**Rapport**

**Réalisé par :** Loussaoui Asmae & Harchan hassan

**Professeurs :** Pr.Mahmoudi

**Année universitaire 2020-2021**

# Introduction

Pour la petite histoire, le NLP commence à se développer dans les années 1950. D'abord sous l'impulsion de chercheurs russes, puis aux États-Unis.

Le NLP (Natural Language Processing ou Traitement Numérique du Langage ) est une discipline qui porte essentiellement sur la compréhension, la manipulation et la génération du langage naturel par les machines. Ainsi, le NLP est réellement à l'interface entre la science informatique et la linguistique. Il porte donc sur la capacité de la machine à interagir directement avec l'humain .il est un terme assez générique qui recouvre un champ d'application très vaste. parmi ces applications les plus populaires : Traduction automatique ,Sentiment analysis et Chatbots ..

et dans notre projet on va se focaliser sur Sentiment analysis , en effet L'analyse des sentiments est devenue un domaine florissant de l'exploration de texte et du traitement du langage naturel. L'analyse des sentiments vise à déterminer si le texte est écrit pour exprimer des émotions positives, négatives ou neutres sur un certain domaine. La plupart des chercheurs en analyse de sentiments se concentrent sur les textes anglais, les ressources disponibles pour d'autres langues complexes, comme l'arabe, étant très limitées . Dans cette étude, l'objectif c'est de développer un modèle initial qui fonctionne de manière satisfaisante et qui mesure les sentiments en français . et pour traiter cette dernière application , il faut utiliser python . étant Python, en plus de son incroyable lisibilité, dispose de bibliothèques remarquables. Dont l'un est NLTK. NLTK ou Natural Language Tool Kit est l'une des meilleures bibliothèques Python NLP. La fonctionnalité qu'il laisse à portée de main tout en conservant sa facilité d'utilisation et encore une fois, la lisibilité est tout simplement fantastique , nous devons mettre la main sur les bibliothèques dont nous avons besoin comme Pandas , Scikit-learn ..

## 1.1

### Les bibliothèques utiliser dans notre projet

- **PyTorch** : On va utiliser la célèbre bibliothèque de deep learning pour l'entraînement de notre modèle.
- **Transformers** : Cette bibliothèque permet de télécharger des versions de camemBERT pré-entraîné. Elle offre même un modèle de classification avec CamemBERT tout-fait.
- **Pandas** : Pour charger notre jeu de données CSV.
- **Scikit-Learn** : Pour évaluer notre modèle grâce aux fonctions de son module metrics.
- **keraas** : bibliothèque permet d'interagir avec les algorithmes de réseaux de neurones profonds et d'apprentissage automatique.
- **nltk** : La plateforme NLTK fournit des interfaces accessibles à plus de cinquante corpus et sources lexicales mis en correspondance avec des algorithmes d'apprentissage automatique, ainsi qu'un choix robuste d'analyseurs syntaxiques et d'utilitaires.

## 1.2

### A propos de notre projet

l'objectif de notre projet c'est de créer une application sentiment analysis qui permet de détecter la nature de sentiment des clients, on va développer notre application avec deux façon différente :

- la première consiste à développer notre propre modèle() .
- la deuxième se base sur l'utilisation d'un transformer qui contient plusieurs modèle pré-entraîné et sentence pièce a le rôle de precessing texte.

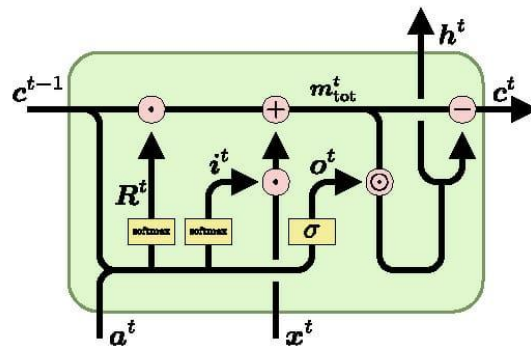
## 1.3

## Modèle LSTM

Long short term memory (LSTM) est une architecture de réseau de neurones récurrents (RNN) utilisé dans le domaine de l'apprentissage en profondeur (deep learning). Une unité LSTM de base est composée d'une cellule, d'une porte d'entrée, d'une porte de sortie et d'une porte d'oubli.

le fonctionnement global d'un LSTM peut se résumer en 3 étapes :

- Détecter les informations pertinentes venant du passé, piochées dans le cell state à travers la Forget gate .
- Choisir, à partir de l'entrée courante, celles qui seront pertinentes à long terme, via l'input gate. Celles-ci seront ajoutées au cell state qui fait office de mémoire longue .
- Piocher dans le nouveau cell state les informations importantes à court terme pour générer le hidden state suivant à travers l'output gate.



## 1.4

## Modèle Camembert

- Camembert est un modèle de langue de pointe pour le français basé sur l'architecture Roberta pré-entraînée sur le sous-corpus français, Camembert améliore l'état de l'art par rapport aux approches monolingues et multilingues précédentes, ce qui confirme l'efficacité des grands modèles de langage pré-entraînés pour le français. Il est une "version" de Roberta pré-entraînée sur un jeu de données francophone. Roberta lui-même est une version de BERT pour laquelle, certains hyperparamètres du pré-entraînement ont été modifiés et l'objectif de prédiction de phrase suivante (Next-Sentence Prédiction) a été supprimé. Camembert hérite donc des avantages de BERT.

## methodology

## 2.1

## Utilisation de modèle LSTM

- Etape 1 : import dataset (qui se compose par 16000 lignes ) à partir de drive pour entrainer le modèle , elle contienne 4 variables : l'id de film ,film url ,commentaires et la nature de sentiments (positifs : 1 ou négatif : 0) .

```
dataset = pd.read_csv("/content/train.csv")
dataset.head(5)
```

	Unnamed: 0	film-url	review	polarity
0	0	http://www.allocine.fr/film/fichefilm-135259/c...	Si vous cherchez du cinéma abrutissant à tous ...	0
1	1	http://www.allocine.fr/film/fichefilm-172430/c...	Trash, re-trash et re-re-trash...! Une horreur...	0
2	2	http://www.allocine.fr/film/fichefilm-15105/cr...	Et si, dans les 5 premières minutes du film, l...	0
3	3	http://www.allocine.fr/film/fichefilm-188629/c...	Mon dieu ! Quelle métaphore filée ! Je suis ab...	0
4	4	http://www.allocine.fr/film/fichefilm-23514/cr...	Premier film de la saga Kozure Okami, "Le Sabr...	1

- **Etape 2 :** créer 2 variables : - reviews qui contient les commentaires et les accorder dans une liste a l'aide de la fonction tolist().  
-Sentiment qui contient les polarity et les accorder dans une liste a l'aide de la fonction tolist().

```
reviews = dataset['review'].values.tolist()
sentiments = dataset['polarity'].values.tolist()
```

- **Etape 3 :** transformer notre dataset de commentaire a une séquence d'entiers , La bibliothèque keras , tokenize() fournit un analyseur lexical L'analyseur de ce module renvoie également des jetons en donne les commentaires a ce analyseur et par la méthode `texts_to_sequences()` en Transforme chaque commantaire a une séquence d'entiers.

```
max_fatures = 512
tokenizer = Tokenizer(num_words=max_fatures, split=' ')
tokenizer.fit_on_texts(reviews)
X = tokenizer.texts_to_sequences(reviews)
X = pad_sequences(X)

sentiments = torch.tensor(sentiments)

sentiments
y = pd.get_dummies(sentiments)
```

- Exemple d'encodage : le traitement se fait selon le dictionnaire .



```
Epoch 1/5
112/112 [=====] - 153s 1s/step - loss: 0.4969 - accuracy: 0.7492
Epoch 2/5
112/112 [=====] - 150s 1s/step - loss: 0.3501 - accuracy: 0.8477
Epoch 3/5
112/112 [=====] - 150s 1s/step - loss: 0.3331 - accuracy: 0.8544
Epoch 4/5
112/112 [=====] - 150s 1s/step - loss: 0.3214 - accuracy: 0.8602
Epoch 5/5
112/112 [=====] - 150s 1s/step - loss: 0.3064 - accuracy: 0.8666
<keras.callbacks.History at 0x7fd348150a10>
```

## 2.2

## Utilisation d'un transformer (CamemBERT) :

- **Etape 1 :** Install la bibliothèque Transformers de HuggingFace et sentencepiece qui a le rôle de processing text() .

```
!pip install flask-ngrok
!pip install flask==0.12.2
!pip install transformers>=4.0
!pip install sentencepiece
```

- **Etape 2 :** unziper notre rar dans lequel on trouve le code html (index.html et test.html) :

```
templates
├── index.html
└── test.html
```

- **Etape 3 :** dans cette étape on va utiliser le modèle `tblard/tf-allocine` ( est un modèle de traitement du langage naturel (NLP) implémenté dans la bibliothèque Transformer, utilisant généralement le langage de programmation Python.) et pipeline qui va prend le modèle et task demandé (sentiment analysis).

```
from transformers import AutoTokenizer, TFAutoModelForSequenceClassification
from transformers import pipeline


tokenizer = AutoTokenizer.from_pretrained("tblard/tf-allocine")
model = TFAutoModelForSequenceClassification.from_pretrained("tblard/tf-allocine")

nlp = pipeline('sentiment-analysis', model=model, tokenizer=tokenizer)
```

- **Etape 4 :** Pour web application et pour aller vite, Flask est un Framework de développement web en Python. Il en existe d'autres, le plus connu d'entre eux est Django mais avec Flask, on peut déjà rapidement obtenir des résultats, sans trop se perdre , dans Ce code en commence par importer le module Flask .

```
from flask import Flask, request, render_template
from flask_ngrok import run_with_ngrok
```

- **Etape 5 :** On donne ensuite un nom à l'application ici ce sera app

```
✓  app = Flask(__name__)  
run_with_ngrok(app) # Start ngrok when app is run
```

- [Etape 6 :](#) définir une page (ou route) avec flask (@app.route) permet de préciser à quelle adresse ce qui suit va s'appliquer. Ici comme on est sur la page d'accueil, on met juste ("/").  
1- Cette exécution qui nous permette de se diriger vers index .html

```
@app.route('/')  
  
def my_form():  
    return render_template('index.html')
```



- 2- Et cette exécution qui va nous permettre de se diriger vers la page test , text1 c'est un var dans lequel la valeur de message sera écrite par l'utilisateur et se stocker en majuscule , et on stock la résultat de nlp dans var a qui va se divise en 2 : text et score



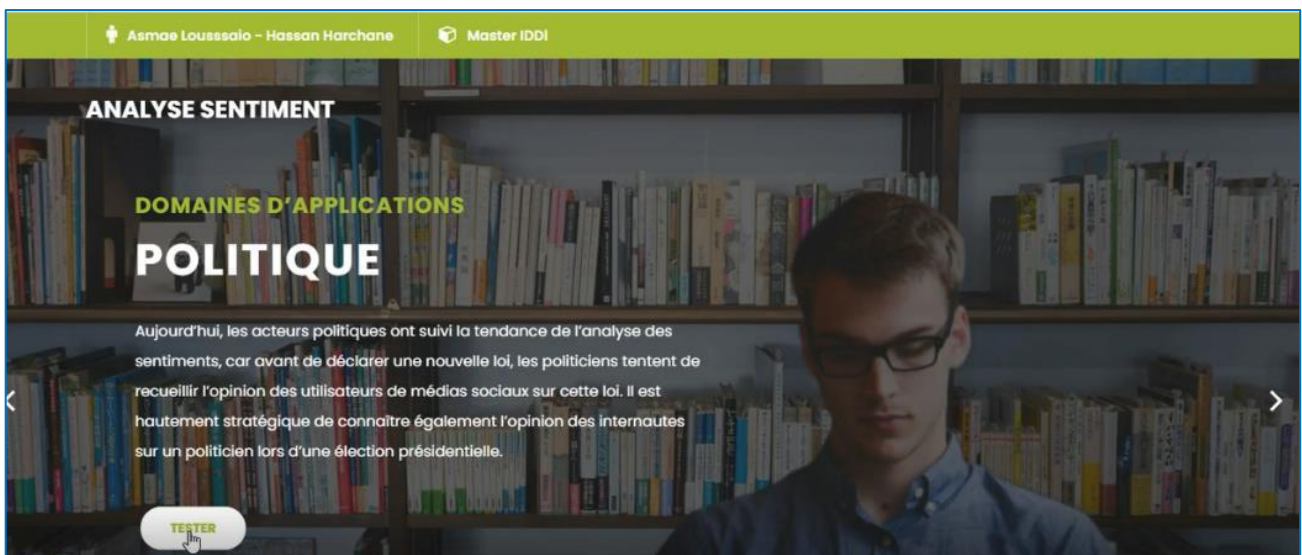
```
@app.route('/', methods=['POST'])
def my_form_post():
    text1 = request.form['message'].lower()

    a = nlp(text1)
    text = a[0]['label']
    score = a[0]['score']
    score = round(score,6)
    if text == 'NEGATIVE' :
        couleur = 'red'
    else :
        couleur = 'green'

    return render_template('test.html', scor=score, final=text, couleur=couleur)

if __name__ == "__main__":
    app.run()
```

\* Running on <http://127.0.0.1:5000/> (Press CTRL+C to quit)  
 \* Running on <http://d547-35-238-27-227.ngrok.io>



- 3- Comme étape finale : c'est la résultat après l'entrer d'un commentaire en français (figure 1) . et d'après click sur Button tester on obtenir la résultat dans figure 2 :



Figure 1 :



Figure 2

