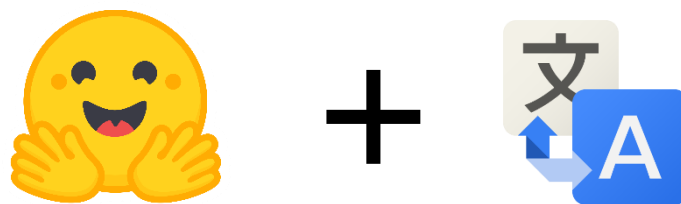




**Master Ingénierie de Données et Développement
Logiciel (Temps Aménagé)**

Module : Natural Language Processing

Mise en place d'une solution de traduction avec Hugging Face



Réalisé Par :

Mouaad Abdelghafour AIT ALI

Radouane DARHOUS

Année universitaire 2020 – 2021

Contenu :

I. Introduction

1. Natural Language Processing (NLP)
2. Hugging Face
3. Transformers

II. Traduction

1. Introduction
2. Comment fonctionne le processus de traduction?
3. Traduction avec Transformers

III. Langages de programmation, technologies et logiciels utilisés

1. Langages de programmation
2. Technologies
3. Logiciels

IV. Démonstration

1. Comment ça fonctionne
2. Front End
3. Back End

V. Conclusion

I. Introduction :

1. Natural Language Processing (NLP) :

Aujourd'hui, le Natural Language Processing couvre de nombreuses tâches comme la compréhension de textes, la classification de sujets, la traduction et bien d'autres. Les chatbots, ces interlocuteurs intelligents, sont simplement la partie émergée de l'iceberg.

Le NLP peut vous aider à extraire l'information que vous ne pourriez pas trouver seul dans un large volume de données textuelles. Il peut contribuer à identifier des bugs rapidement en lisant les commentaires sur un produit. Les options sont infinies et le futur est rempli d'améliorations encore plus sensationnelles ! Pour l'instant la génération de texte est l'un des challenges majeurs dans ce domaine.

Malgré ces impressionnants progrès, il subsiste néanmoins quelques limites. Pour l'instant cela nécessite un gros volume de données. La plupart du temps, elles proviennent d'Internet, mais la qualité n'est pas forcément au rendez-vous. De plus, Internet n'est pas comparable à la vraie vie de tous les jours et des biais peuvent survenir dans les décisions de l'IA. Et entraîner ces intelligences artificielles coûte beaucoup d'argent, car cela nécessite d'énormes fermes de serveurs pour obtenir des résultats satisfaisants.

2. Hugging Face :

Hugging Face 🤗 est un fournisseur open source de technologies de Natural Language Processing (NLP). Nous pouvons utiliser des modèles à la pointe de la technologie (sous la bibliothèque Transformers) pour créer et entraîner nos propres modèles. Nous pouvons utiliser la bibliothèque d'ensembles de données de visage étreignant pour partager et charger des ensembles de données. De même, nous pouvons même utiliser cette bibliothèque pour les métriques d'évaluation.

3. Transformers :

Cette bibliothèque fournit des milliers de modèles qui permettent à un développeur d'effectuer diverses tâches NLP. Quelques-uns incluent la classification de texte, la recherche d'informations, l'extraction d'informations, le résumé abstrait et extractif, la reconnaissance d'entités de nom, l'inférence de langage naturel, la traduction de texte, la génération de texte, la réponse à des questions, le sous-titrage d'images, etc. pour n'en nommer que quelques-uns.

II. Traduction :

1. Introduction :

En utilisant la NLP, nous pouvons créer un système de traduction pour nous conduire vers une communication ouverte et efficace. La capacité émergente des ordinateurs à comprendre et à analyser le langage humain est le Natural Language Processing.

Pour que l'ordinateur comprenne les mots que nous utilisons dans nos langues, nous décomposons les paragraphes et les phrases en unités de langue. Ces unités sont converties en nombres puis de nouveau en mots, mais ceux d'une langue différente, complétant ainsi le processus de traduction. Il y a un certain nombre de concepts et d'algorithmes impliqués dans ce processus, que nous allons parcourir dans cette histoire :

- **Preprocessing** : Tokenisation et remplacement de mot inconnu
- **Word Embeddings** : Vecteurs et réduction de dimension
- **Sequence to Sequence**: Encodage et décodage

2. Comment fonctionne le processus de traduction ?

Le prétraitement se produit au début de la traduction automatique, convertissant le texte en données brutes en une forme que la machine comprendra. Ce processus comporte deux parties principales : la tokenisation et le remplacement des mots inconnus. Le prétraitement peut même consister à convertir les données en minuscules pour que la machine traite les mots en majuscules comme un même mot.

```
In [11]: from nltk.tokenize import word_tokenize
text = "I like reading books"
print(word_tokenize(text))

['I', 'like', 'reading', 'books']
```

- **TOKENISATION**

La tokenisation est la décomposition des données plus volumineuses sous forme de paragraphes et de phrases en « jetons » ou unités de langage. Ces unités sont souvent des mots et des ponctuations individuelles. Deux jetons supplémentaires que nous introduisons marquent le début d'une phrase (SOS) et la fin d'une phrase (EOS).

- **REEMPLACEMENT UNK (UNKNOWN WORDS)**

Lors de l'entraînement de la machine avec des ensembles de données, certains mots n'apparaissent pas suffisamment de fois pour que le programme apprenne leur signification. Ainsi, lorsqu'il ne reconnaît pas ou ne comprend pas un mot, il le remplace par UNK comme dans inconnu, puis poursuit la traduction. Ces lacunes dans les phrases peuvent être comblées plus tard par un post-traitement.

- **WORD EMBEDDINGS**

Après le prétraitement, la traduction automatique utilise des incorporations de mots. Ce sont des représentations vectorielles de textes. Fondamentalement, chaque phrase se voit attribuer une série de nombres, également appelés vecteurs, avec un nombre arbitraire de dimensions. Ces dimensions aident à décrire les mots d'une phrase.

Les vecteurs sont si importants lorsqu'il s'agit d'inclusions de mots, car les valeurs permettent à la langue de changer tandis que leurs significations restent les mêmes. Le traducteur se concentrera sur ces valeurs numériques plutôt que sur des valeurs de chaîne ou des définitions. Les vecteurs deviennent plus intéressants lorsque nous les analysons pour trouver des relations entre les mots. Nous pouvons même utiliser des opérations mathématiques sur eux, y compris l'addition et la soustraction. Cela nous permet d'identifier les similitudes et les différences entre les mots et les phrases.

```
In [21]: from gensim.models.keyedvectors import KeyedVectors
# from gensim.test.

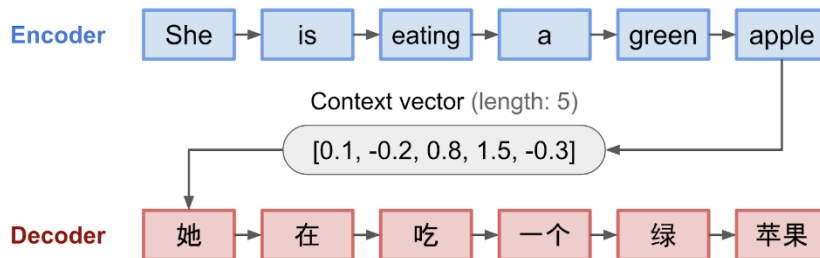
model = KeyedVectors.load_word2vec_format("https://s3.amazonaws.com/dl4j-distribution/GoogleNews-vectors-negative300.bin.gz",
                                          binary=True, limit=30000)
print(model.most_similar(positive=['king', 'women'], negative=['man'], topn=5))

[('queen', 0.4827326238155365), ('kings', 0.4558638632297516), ('monarch', 0.39624831080436707), ('monarchy', 0.39430150389671326), ('Women', 0.38392189145088196)]
```

Par exemple dans l'équation suivante : **King — Man + Women = Queen**, nous comprenons que le sens d'une reine ressemble plus à un Roi et une Femme qu'à un Homme.

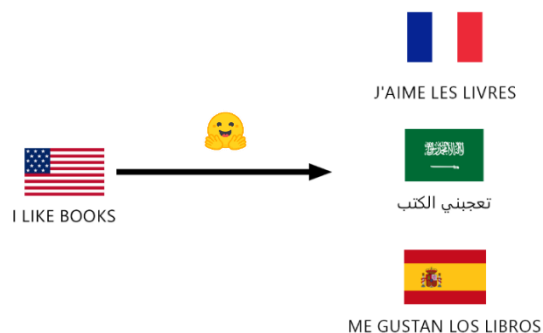
- **SEQUENCE TO SEQUENCE**

Le système qui utilise des vecteurs pour traduire une phrase ou une séquence de mots est appelé séquence à séquence.



Il est composé d'un encodeur et d'un décodeur. L'encodage permet au programme de convertir la phrase originale en un vecteur pour la compréhension. C'est un peu comme décrire l'apparence du coupable à un dessinateur. Ensuite, le décodage transforme les valeurs du vecteur en langage humain, mais cette fois dans le nouveau langage dans lequel nous voulions les traduire.

3. Traduction avec Transformers :



La bibliothèque Transformers permet aux contributeurs de publier des ensembles de données linguistiques et de partager des modèles pré-entraînés. En mai 2020, le Language Technology Research Group de l'Université d'Helsinki (**Helsinki-NLP**) a publié un grand nombre de modèles de traduction dans la bibliothèque Transformers. Ces modèles ont été entraînés à l'aide du cadre **AutoModelWithLMHead** et de l'ensemble de données Open Parallel Corpus (OPUS). L'ensemble de modèles comprend plus de 1000 paires de langues et 169 traductions de langues source ou de familles de langues vers l'anglais.

L'utilisation de ces modèles pré-entraînés est simple. L'exemple ci-dessous montre comment charger les modèles dynamiquement à partir du hub de modèles Hugging Face, ou vous pouvez les télécharger et les enregistrer localement pour y accéder hors ligne.

III. Langages de programmation, technologies et logiciels utilisés :

1. Langages de programmation :



Python : est un langage de programmation polyvalent, polyvalent et puissant. C'est une excellente première langue parce qu'elle est concise et facile à lire. Quoi que vous vouliez faire, Python peut le faire. Du développement Web à l'apprentissage automatique en passant par la science des données, Python est le langage qu'il vous faut.

- **Cas d'utilisation** : pour répondre à notre besoin de traduire un texte dans plusieurs langues à l'aide de HuggingFace Transformers



Kotlin : est un langage de programmation multiplateforme, à typage statique et à usage général avec inférence de type. Kotlin est conçu pour interagir pleinement avec Java, et la version JVM de la bibliothèque standard de Kotlin dépend de la bibliothèque de classes Java, mais l'inférence de type permet à sa syntaxe d'être plus concise.

- **Cas d'utilisation** : pour développer l'application Android (front end de notre projet)



Java est un langage de programmation orienté objet de haut niveau, basé sur des classes, conçu pour avoir le moins de dépendances d'implémentation possible.

- **Cas d'utilisation** : pour développer l'application Desktop (Back end de notre projet)

2. Technologies :



Firestore : La base de données Firebase Realtime est une base de données NoSQL hébergée dans le cloud. Les données sont stockées au format JSON et synchronisées en temps réel avec chaque client connecté. Lorsque vous créez des applications multiplateformes avec SDK iOS, Android et JavaScript, tous vos clients partagent une instance de base de données en temps réel et reçoivent automatiquement des mises à jour avec les données les plus récentes.

Cas d'utilisation : pour enregistrer dans la base de données le texte saisi par l'utilisateur depuis l'application Android (front end), afin que l'application de bureau (back end) puisse l'obtenir et commencer à exécuter le script python afin de traduire ce texte dans la langue cible.

3. Logiciels :



Android Studio : est l'environnement de développement intégré (IDE) officiel pour le système d'exploitation Android de Google, construit sur le logiciel IntelliJ IDEA de JetBrains et conçu spécifiquement pour le développement Android. Il est disponible en téléchargement sur les systèmes d'exploitation Windows, macOS et Linux ou en tant que service par abonnement en 2020. Il remplace les outils de développement Android Eclipse en tant qu'IDE principal pour le développement d'applications Android natives.

Cas d'utilisation : Android Studio est notre environnement de développement intégré pour créer l'application Android (Front end).



NetBeans : est un environnement de développement intégré (IDE) pour Java. NetBeans permet de développer des applications à partir d'un ensemble de composants logiciels modulaires appelés modules. NetBeans fonctionne sous Windows, macOS, Linux et Solaris. En plus du développement Java, il possède des extensions pour d'autres langages comme PHP, C, C++, HTML5 et JavaScript. Les applications basées sur NetBeans, y compris l'IDE NetBeans, peuvent être étendues par des développeurs tiers.

Cas d'utilisation : NetBeans est notre environnement de développement intégré pour créer l'application bureau (Back end).

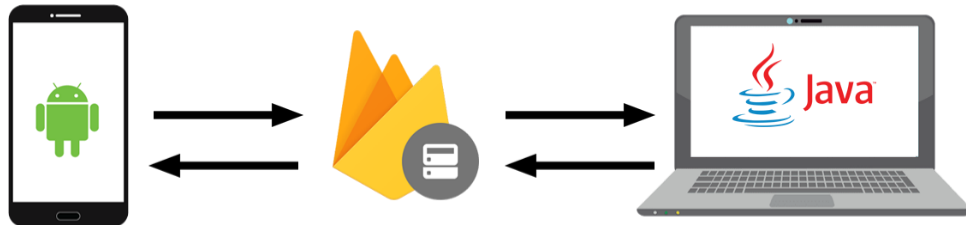
IV. Démonstration :

1. Comment ça fonctionne :

1. L'utilisateur entre un texte et sélectionne la langue du texte et la langue vers laquelle traduire.

2. Le texte saisi ainsi que les langues d'entrée et de sortie sont enregistrés dans la base de données Firebase.

3. L'application Java reçoit le texte de la base de données et commence à exécuter le script python.



4. Lorsque le script python a fini de s'exécuter, le texte traduit est enregistré dans la base de données, afin que l'application Android puisse le recevoir.

2. Front end:

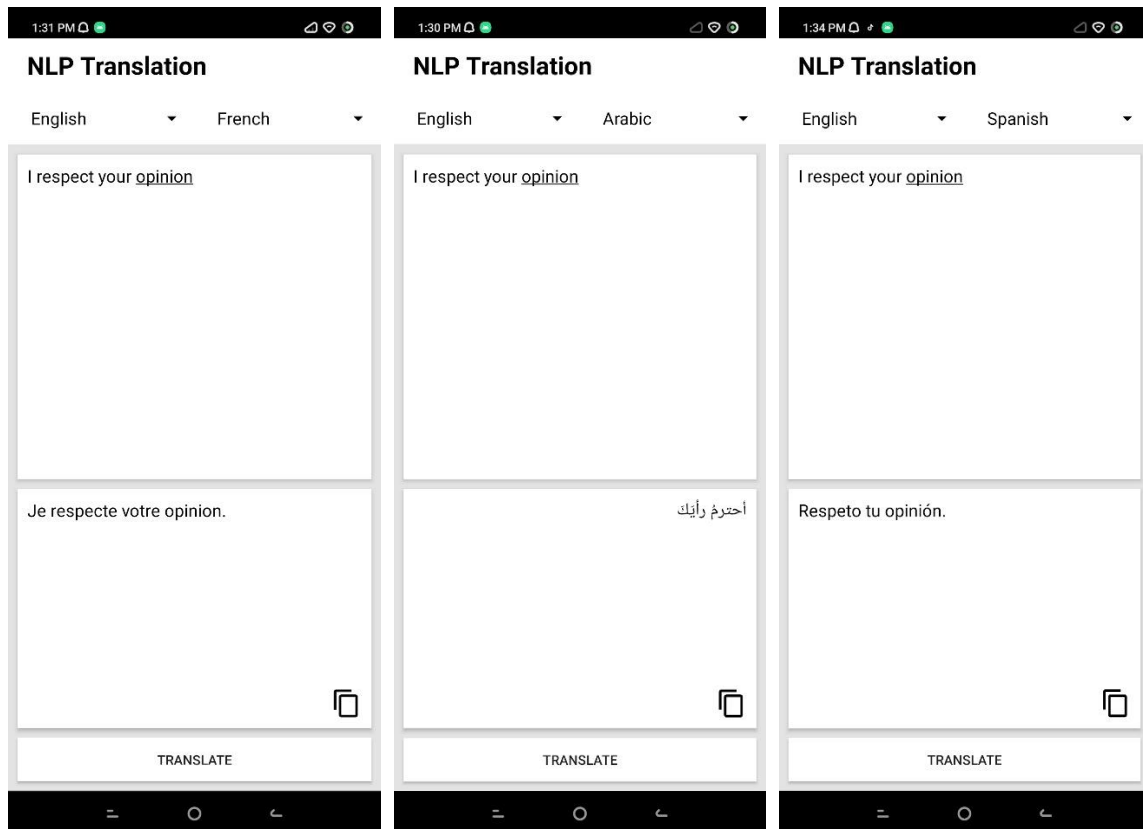
Nous avons développé une application Android utilisant Kotlin, l'application permet à l'utilisateur de saisir du texte et de sélectionner la langue dans laquelle traduire.



Lorsque l'utilisateur clique sur le bouton "**TRANSLATE**", le texte saisi est enregistré dans la base de données Firebase avec la langue du texte et la langue cible comme ceci :


INPUT_LANG \$ **TEXT** \$ **OUTPUT_LANG**

Si tout se passe bien, l'application de bureau commence à exécuter le script python avec le texte enregistré.



Remarque : Nous pouvons également traduire de (arabe, français ou espagnol) vers l'anglais, en général, nous pouvons traduire de n'importe quelle langue X vers la langue Y.

Lorsque l'exécution du script python est terminée, le texte traduit est enregistré dans la base de données et s'affiche automatiquement dans l'application Android grâce au processus de synchronisation Firebase.

En cliquant simplement sur le bouton de copie  les utilisateurs peuvent copier le texte traduit s'ils le souhaitent.

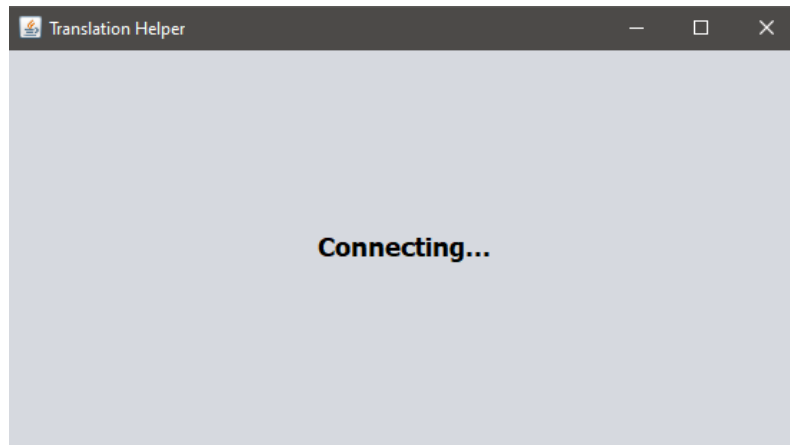
Remarque : En raison de la grande taille des modèles Helsinki, le processus de traduction prend un certain temps (> 10 secondes) pour se terminer.

3. Back end:

Comme on le sait, Android ne peut pas exécuter le script python, nous avons donc développé une application de bureau Java pour ce travail.

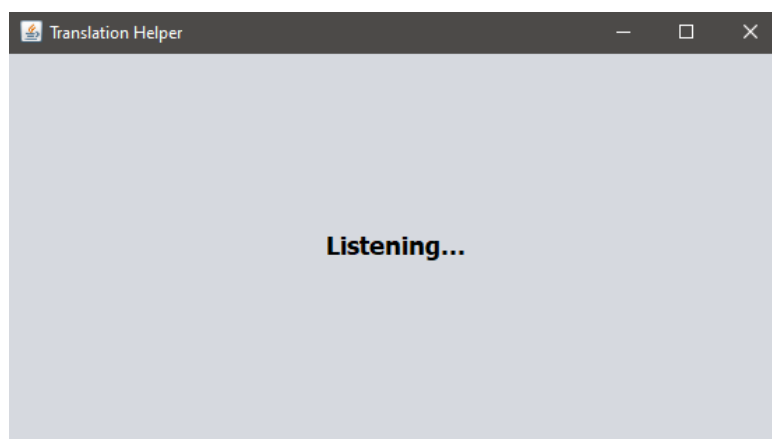
L'application Java affiche 3 états comme suit :

1. Connecting



Au moment de l'exécution, l'application commence à se connecter à la base de données Firebase.

2. Listening



Après la connexion réussie à la base de données Firebase, l'application commence à écouter tout processus de traduction possible.

3. Executing python script



Lorsque l'utilisateur saisit un texte et clique sur le bouton "**TRANSLATE**" dans l'application Android, l'application java passe automatiquement au troisième état, qui exécute le script python.

Cet état signifie que l'application Java a reçu le texte de la base de données ainsi que les langages d'entrée et de sortie, l'application java extrait et remplace les éléments suivants :

INPUT_LANG : la langue du texte saisi.

TEXTE : le texte saisi par l'utilisateur à partir de l'application Android.

OUTPUT_LANG : la langue cible vers laquelle l'utilisateur souhaite traduire.

```
private String generateInput(String input) {
    String[] inputs = input.split("\\$");

    String isoInputLang = inputs[0];
    String textInput = inputs[1];

    String isoOutputLang = inputs[2];
    String mname = "Helsinki-NLP/opus-mt-" + isoInputLang + "-" + isoOutputLang;
    String fullScript = "# -*- coding: utf-8 -*-\n"
        + "import sys\n"
        + "from transformers import AutoModelWithLMHead, AutoTokenizer\n"
        + "from transformers import pipeline\n"
        + "\n"
        + "\n"
        + "mname = " + "\"" + mname + "\"\n"
        + "\n"
        + "model = AutoModelWithLMHead.from_pretrained(mname)\n"
        + "tokenizer = AutoTokenizer.from_pretrained(mname)\n"
        + "translation = pipeline(\"translation_\" + isoInputLang + \"_to_\" + isoOutputLang + "\", model=model, tokenizer=tokenizer)\n"
        + "text = " + "\"" + textInput + "\"\n"
        + "\n"
        + "translated_text = translation(text, max_length=40)[0]['translation_text']\n"
        + "\n"
        + "sys.stdout.buffer.write(translated_text.encode(\"utf-8\"));
    System.out.println("==> " + fullScript);
    return fullScript;
}
```

A diagram illustrating the variable substitution in the Python script generation code. Green arrows point from the variable names in the code to their corresponding values: 'isoInputLang' points to 'fr', 'isoOutputLang' points to 'en', and 'mname' points to 'Helsinki-NLP/opus-mt-fr-en'. A red arrow points from 'textInput' to the text 'Hello world'. A blue arrow points from the 'translation' variable to the 'translation' key in the dictionary output. The final line of the script, which prints the translated text, is highlighted in yellow.

Si l'exécution se termine, l'application Java revient automatiquement à l'état précédent, à savoir "**Listening**" pour le prochain processus de traduction.

V. Conclusion :

Les programmes de traduction créent de nombreuses opportunités de communication, utilisées avec les touristes et de nombreuses entreprises. Avec des programmes améliorés, la diffusion des connaissances et des informations dans le monde entier n'aura peut-être plus besoin d'être aussi coûteuse ou chronophage. Inutile de dire que le traitement du langage naturel est un système étendu qui a la capacité de contribuer beaucoup à notre société moderne.