Course: COSC 4337 Data Science II

Professor: Ricardo Vilalta

TA: Shaila Zaman

Team : 4

Group members: Hieu Trinh , Thanh Le

**Machine Translation from English to French**

**Data description :**

**Source:** This dataset is created based on the website : <http://www.manythings.org/anki/> , which collects a lot of datasets of different language. The author only used sentences that were owned by identified native speakers working on the Tatoeba Project and English sentences that he personally checked and did not reject.

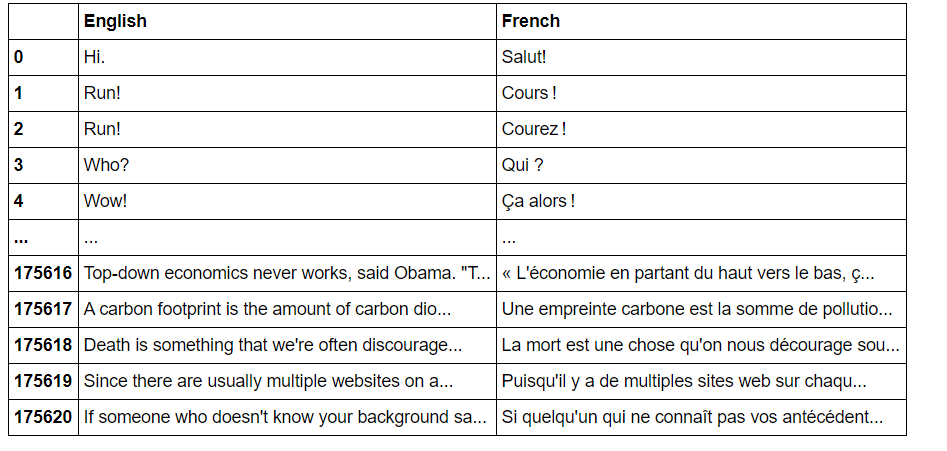
**Dataset overview:** The dataset has two columns : one has English words/sentences and the other has French words/sentences. There are 175621 rows for both columns as the English words/sentences are matched with the French words/sentences for each row, so this is a balanced dataset. The special thing about this dataset is that one English word could be represented in different ways in French, so some words could be repeated in the dataset. For example, “Run!” could be represented as “Cours !” or “Courez !”. There are no null values for both columns, and as we discover along throughout the report, especially when looking at the Exploratory Data Analysis Notebook, we will have a more insightful look of the dataset.

**Project goal:** We will build a deep neural network that functions as part of a machine translation pipeline that takes input of English texts and returns French translation. The goal is to achieve the highest translation accuracy by building different models.

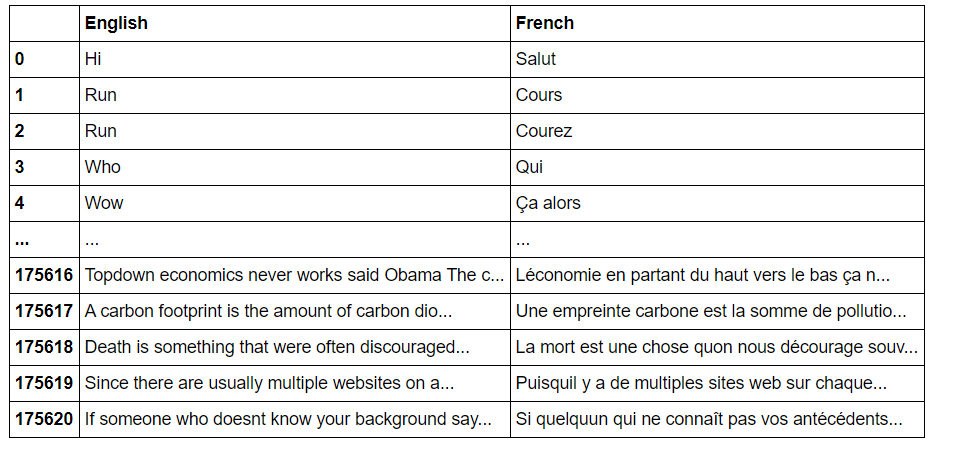
**Exploratory Data Analysis & Feature Engineering:**

**Import Dataset and Library**

To begin with the preprocessing step, we first import Pandas and built-in functions needed to read the dataset. The dataset has 2 columns with 175621 rows. We then simply rename two columns as “English” and “French” to make it easier to call.

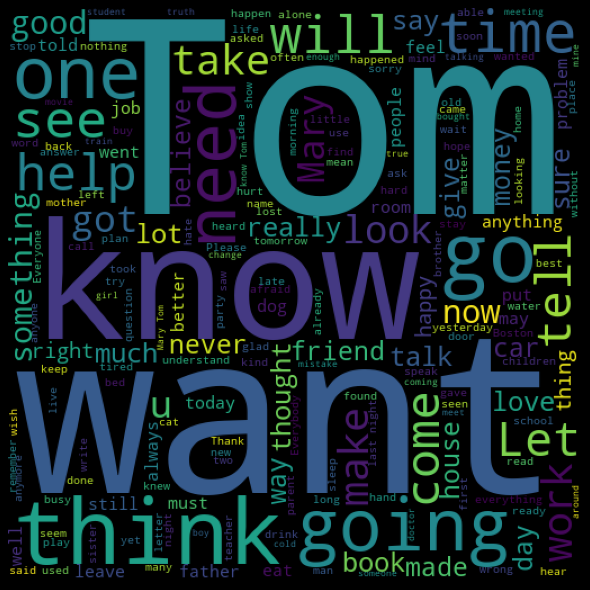


**Cleaning the Dataset:**

This is a machine translation task, so we do not want to remove any words, even stop words such as the, a , an, in, etc… or regular expressions. However, we want to have a better look at our dataset because we try to understand the meaning of words , so we should remove all the regular expressions and punctuations before performing data analysis..

**Data Visualization:**

We want to see the frequency of each word, so we use the library wordcloud, which is a data visualization technique used for representing text data in which the size of each word indicate its importance. From the graph, we can clearly see which tokens mostly occur in the dataset.

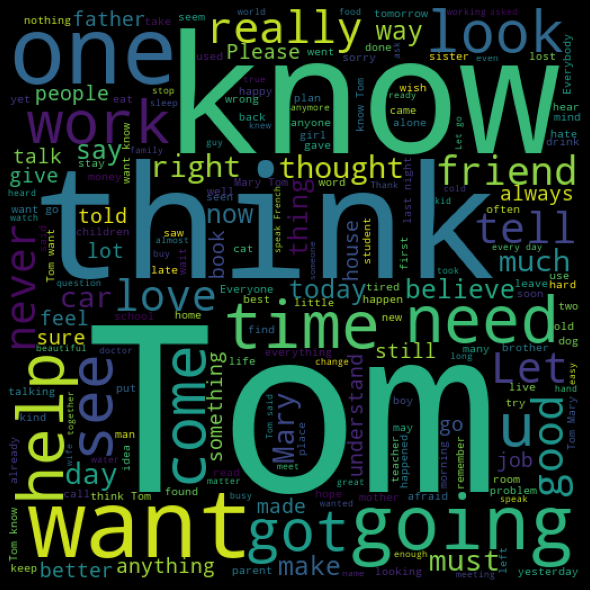
1. **Original dataset:**
2. **English**

As we can see, a lot of familiar words like know, want, think, going, will, etc… occur a lot in the dataset. By using a lot of popular words, it will make it easier for the translation because we do not want to interfere with too many complicated words.

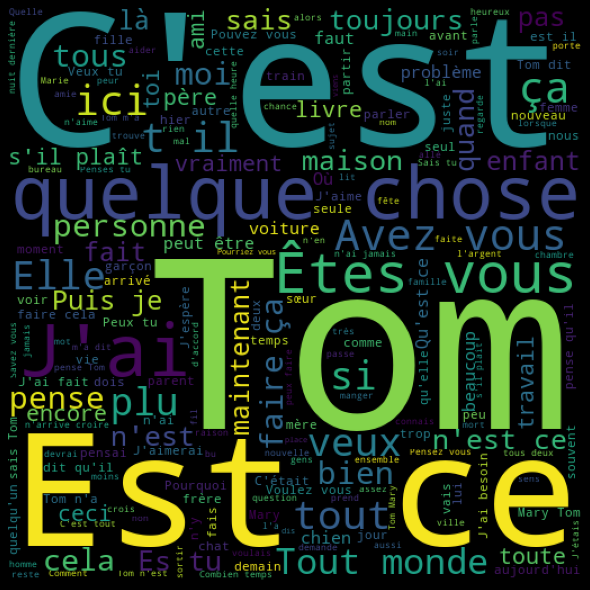
1. **French**

In French, some words like Je, que, suis are very popular because they are translated as I, that, am. It is easy to understand because those tokens also occurs a lot in English.

1. **Dataset after removing stop words**
2. **English**



We do not see a lot of difference comparing to the original one, so we can assume that our English column is pretty clean even before preprocessing.

1. **French**

In opposite to English, we see a lot of changes in French after removing stop words. We see new tokens like C’est , Est , ce and they are translated as It is, is, this. These are also popular words in English.

After performing wordcloud visualization, we go ahead and remove all the regular expressions and punctuations. Then we check if the sentences contain number in English and French column, but we see that sentences from both languages use the numbers but not the similar words like “one” for “1” , so we decide that we will keep numbers in sentences because it will not affect the translation process. Then, we create English and French dictionaries of our dataset to keep track of the frequency of each word.

Also, we want to understand more about how words are distributed throughout sentences, so we decide to plot the top 10 most frequents words in the bar graph for analysis by using seaborn library. We basically use the Counter function from collections library to collect all the unique tokens in English and French.

**Before removing stopwords:**

Figure 1: English words with high frequency

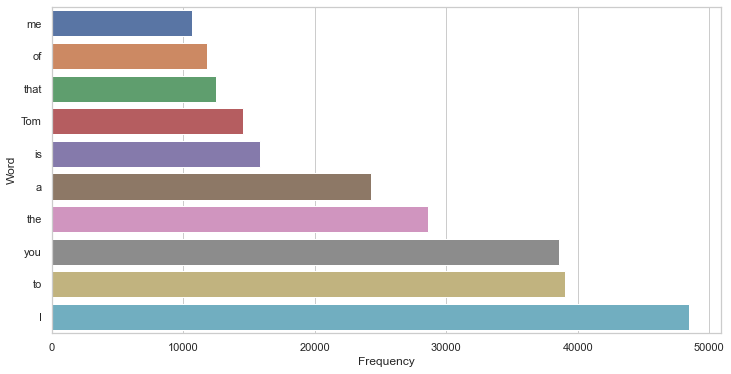
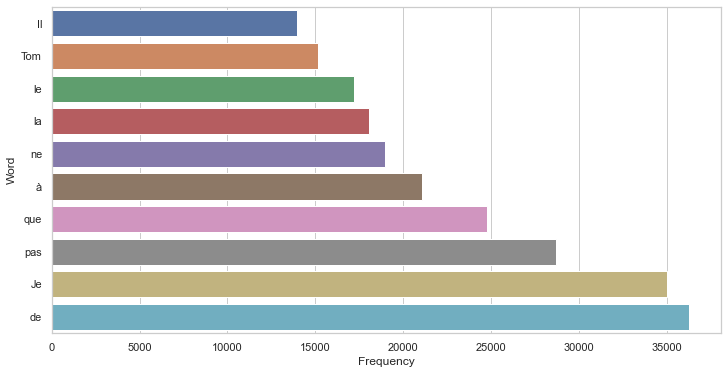


Figure 2: French words with high frequency



**After removing stop words:**

Figure 3: English words with high frequency

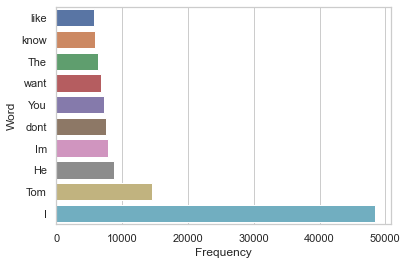
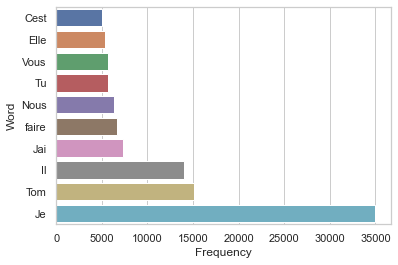
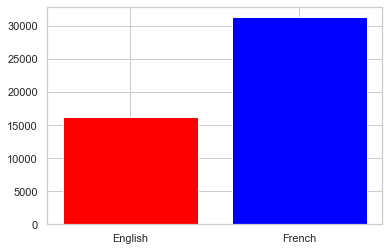


Figure 4: French words with high frequency



The results before and after removing stop words are so different. Before removing, we can clearly see a lot of words like I, to, you, the occur around 30000 – 48000 times in sentences. However, after the removal, the lead automatically belongs to “I” and it shows us that stop words have a great impact on every language, and it occurs almost in every sentence.

Before moving on to discuss about the POS tags, we want to show the uniqueness between two languages:



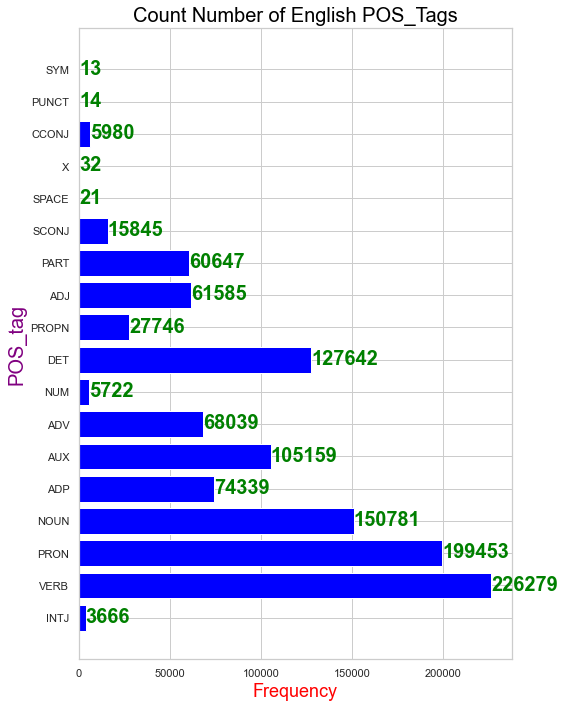
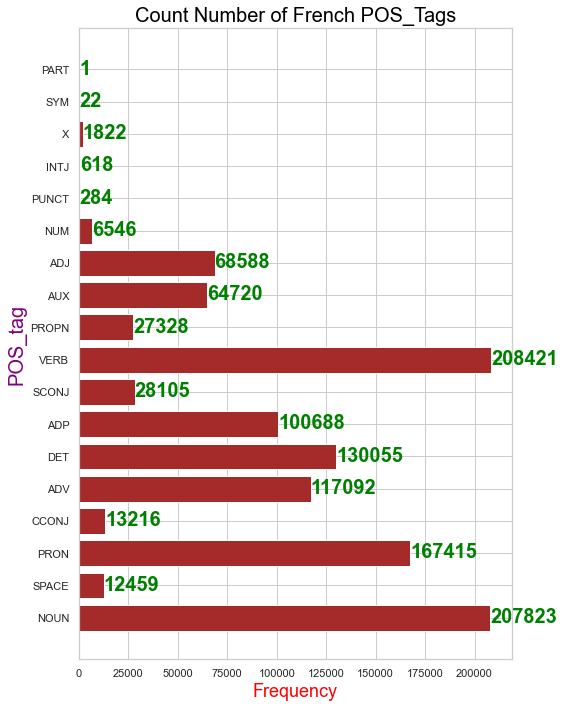
We can easily see that for this particular dataset, French tokens almost double English tokens. This again shows that one English word can be expressed differently in French.

**Understanding Part of Speech Tagging (POS Tag):** POS tag is a special technique to identify the label assigned to each token that we can understand what type of tokens mostly occur in the dataset.

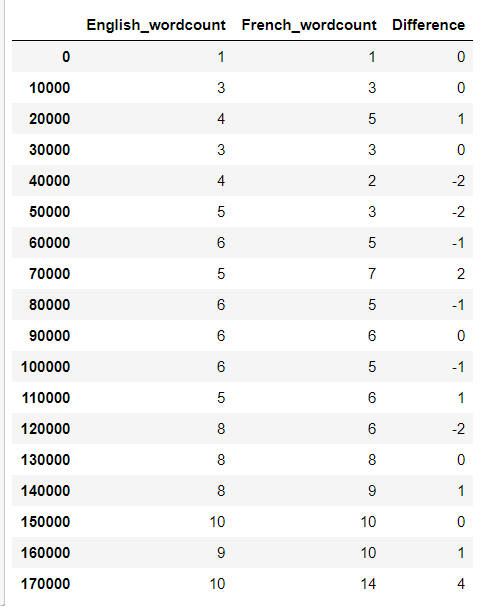
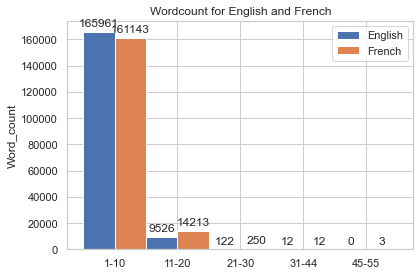
For example: I like you => “like” is a verb with a positive sentiment.

I am like you => “like” is a preposition with a neutral sentiment.

This time, we use the bar analysis from the matplotlib library to visualize the distribution of POS tags:



We use the pre-trained models from spacy to create dictionaries of POS tag for English and French. The results are not surprising when we see a lot of verbs, nouns and pronouns occur a lot in both languages. Once again, it shows the grammar and the structure of sentences between two languages have a lot of similarities.

We want to explore the dataset further, so we count and compare the number of words for each row and try to create different categories depending on the length of sentences. This will help us to find out most sentences fall into which category.

We can see that most sentences from both languages fall into the range of 1 to 10 words. This is extremely helpful because we focus more on the accuracy of translating each word rather than the accuracy of the long and complicated sentences.

**Summary of the dataset :**

1082094 English words.

16414 unique English words.

10 Most common words in the English dataset:

"I" "to" "you" "the" "a" "is" "Tom" "that" "of" "me"

1142744 French words.

31487 unique French words.

10 Most common words in the French dataset:

"de" "Je" "pas" "que" "à" "ne" "la" "le" "Tom" "Il"

**Preprocessing pipeline:**

1. **Tokenization**

As deep learning models cannot understand text, we need to convert text into

numerical representation. Therefore, we need to perform a technique called Tokenization, which split sentences into words and encodes these into integers.

For example:

text\_sentences = [

'The quick brown fox jumps over the lazy dog .',

'I really love my dog .',

'My girlfriend is so beautiful .']

**After tokenization** :

{'the': 1, 'dog': 2, 'my': 3, 'quick': 4, 'brown': 5, 'fox': 6, 'jumps': 7, 'over': 8, 'lazy': 9, 'i': 10, 'really': 11, 'love': 12, 'girlfriend': 13, 'is': 14, 'so': 15, 'beautiful': 16}

Then, we represent these as sequences from the tokenizer object we created

Sequence 1 in

Input: The quick brown fox jumps over the lazy dog .

Output: [1, 4, 5, 6, 7, 8, 1, 9, 2]

Sequence 2 in x

Input: I really love my dog .

Output: [10, 11, 12, 3, 2]

Sequence 3 in x

Input: My girlfriend is so beautiful .

Output: [3, 13, 14, 15, 16]

1. **Padding**

In the dataset, the length of sentences is different. However, neural networks require to have inputs with the same size. For this purpose, padding technique has to be performed.

Sequence 1 in x

Input: [1 4 5 6 7 8 1 9 2]

Output: [1 4 5 6 7 8 1 9 2]

Sequence 2 in x

Input: [10 11 12 3 2]

Output: [10 11 12 3 2 0 0 0 0]

Sequence 3 in x

Input: [ 3 13 14 15 16]

Output: [ 3 13 14 15 16 0 0 0 0]