### ****N-grams and Next Word Prediction****

The objective of this exercise is to predict the next word in a text sequence from a corpus, using the **n-gram technique**. Specifically, we aim to build a simple model that, given a word (or a sequence of words), suggests the most probable following words based on the observed frequencies in a textual corpus.

#### ****Methodology****

We used the **“brown” corpus** from the NLTK library, which contains a diverse set of English texts. The raw text was:

* Tokenized,
* Cleaned (punctuation removed and converted to lowercase),
* Filtered to keep only alphabetic words.

We then applied the n-gram logic to generate consecutive sequences of n words from the tokens. Two main steps were performed:

1. Construction of a dataframe of n-grams with their frequencies;
2. Creation of a function to predict the next word, based on a sequence of n–1 words.

#### ****Technical Steps****

* For n = 2, we generate **bigrams**: for example ("of", "the") or ("in", "the").
* For n = 3, we construct **trigrams** like ("one", "of", "the").

We defined a function get\_ngrams\_frequencies() which returns a DataFrame sorted by descending frequency, allowing us to identify the most frequent combinations.  
Then, a second function predict\_next\_words() suggests the most probable words that follow a given input sequence, according to the chosen n-gram model (bigram, trigram, etc.).

#### ****Observed Results****

* For bigrams, the most frequent combinations are:
  + ("of", "the"), ("in", "the"), ("to", "the"), etc.
  + These results reflect common syntactic co-occurrences in written English.
* Example of prediction with input\_seq = ("in",) and n = 2, k = 5:
  + The model returns: ["the", "a", "his", "this", "which"].
  + These words are grammatically coherent after “in”, showing that the model effectively captures language probabilities.
* For trigrams, sequences like ("one", "of", "the") or ("the", "united", "states") appear with high frequencies.
  + When given ("one", "of") as input, the predictions are consistent: ["the", "best", "most", ...].

#### ****Analysis****

The n-gram approach relies solely on local occurrence frequencies. It is therefore:

* Simple to implement, but
* Highly dependent on local context: the model does not generalize and does not understand meaning, unlike modern embeddings.

However, for tasks such as **automatic word suggestion** or **basic statistical analyses**, this type of model remains very efficient and quick to deploy.