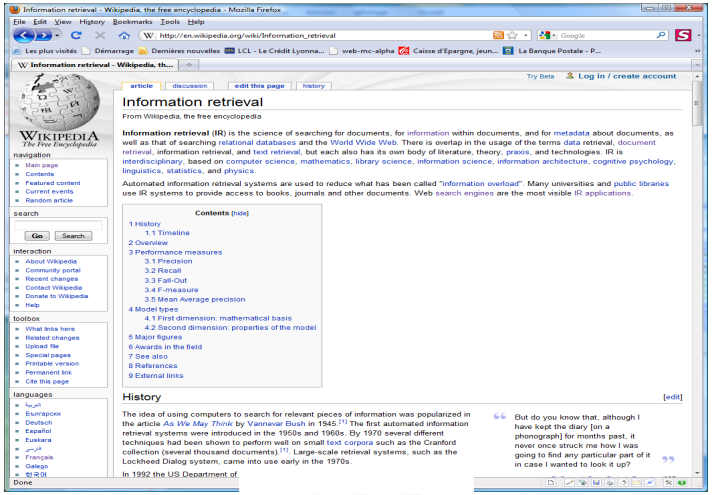
Chapitre 1 : Représentation des textes
Représentation en Bag of words

Représentation de textes

- Opération qui consiste à traduire un texte dans une forme utilisable par un ordinateur
- Opération primordiale pour toutes les tâches qui exploitent le texte

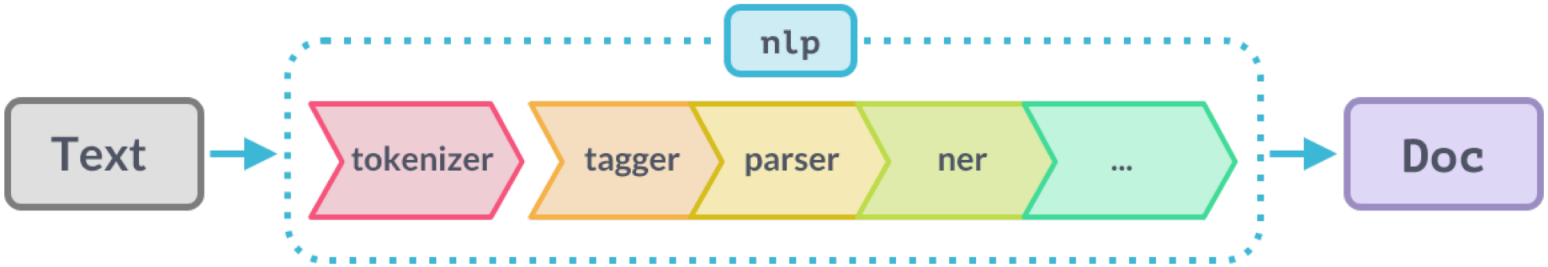
Une bonne majorité de tâches relatives au traitement de textes passe par une phase de représentation du texte → on représente un texte dans une forme facilement exploitable par l'ordinateur



(…., ……,…, …, .., …., ….)

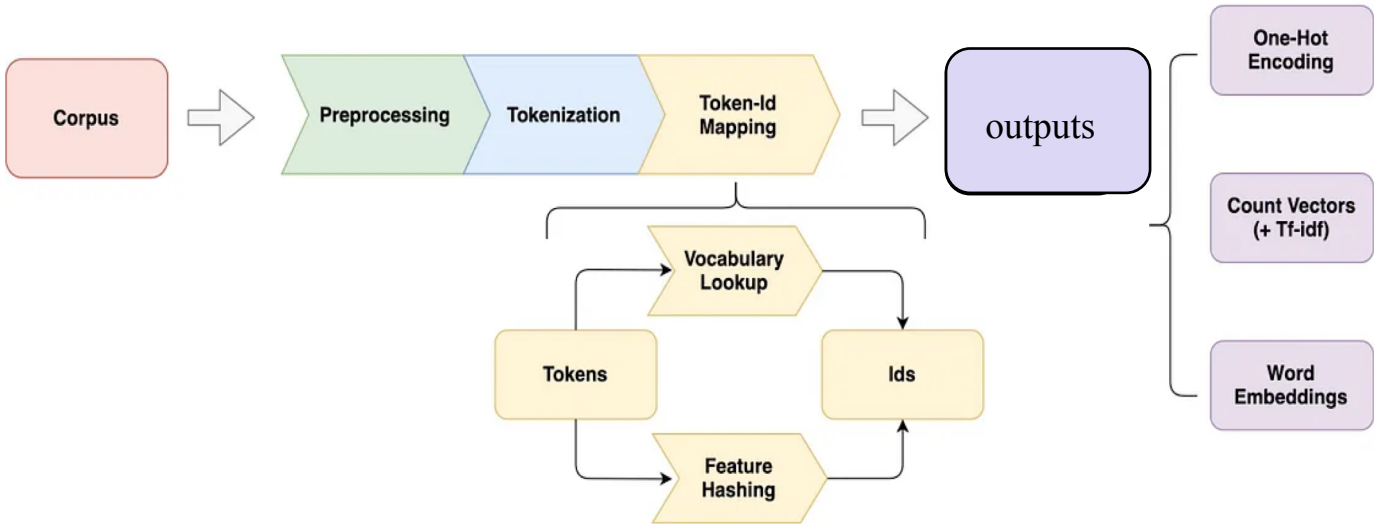
Pipeline *"Text processing"*

- Le pipeline habituel du « *Text processing* »



Pipeline plus “léger”

- Un Pipeline léger utilisé dans « beaucoup » de tâches qui ne demandent pas d’analyse fine (cadres statistiques / machine learning, ..)
- → Limiter à la phase de « ségmentation (tokenization) des textes en mots isolés (tokens) » + la pondération des mots



Représentation textutelle : le pipeline

- Décomposer le texte (Parsing)

<Title>: Information retrieval
(Corps du texte> : Information retrieval (IR) is the science of searching of documents. N.I.S.T launched TREC

- Segmenter les séquences de caractères en mots, en sous mots, …, (Tokenizing)
- Normaliser
  - Textuelle: ponctuation, dates, case
  - Linguistique : Racinisation (stemming)/lemmatisation

Information retrieval Information retrieval IR is the science of searching of documents N.I.S.T launched TREC

information retrieval information retrieval IR is the science, of search, document NIST launched TREC

- Supprimer les mots communs (stop word removal)
  - En fonction d'une "short list" "the", "and", "or", ou mots fréquents

information, retrieval, information, retrieval, IR, science, search, document NIST launch TREC

- Regrouper les mots

information 2, retrieval 2, IR 1, science 1, search 1, document 1, NIST 1, launch 1, TREC 1

- Associer à chaque mot un identifiant (Id) (liste de mots triée)

2 2, **6** 2, **3** 1, **8** 1, **9** 1, **1** 1, **5** 1, **4** 1, **10** 1

Un sac de mots (BOW)

Segmentation (Tokenization)

- Quels séparateurs
  - Pas d'espaces en chinois et en japonais
    - Ne garantit pas l'extraction d'un terme de manière unique

Chinese tokenizer

1. Original text

旱灾在中国造成的影响

(the impact of droughts in China)

2. Word segmentation

旱灾 在 中国 造成 的 影响

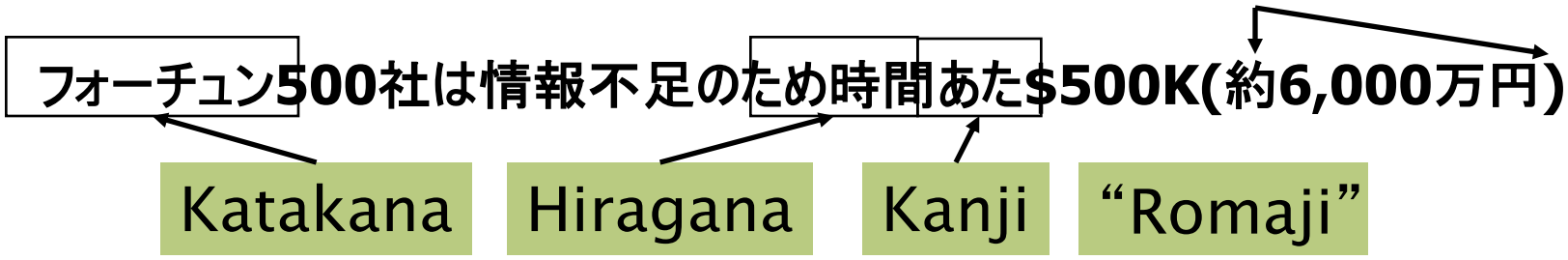
drought at china make impact

3. Bigrams

旱灾 灾在 在中 中国 国造
造成 成的 的影 影响

Segmentation (Tokenization)

- Japonais encore plus compliqué avec différents alphabets



L’utilisateur peut exprimer sa requête entièrement en Hiragana

Normalisation

- Processus morphologique permettant de regrouper les variantes d'un mot
  - Lemmatisation : analyse linguistique pour ramener les mots à leur lemme (verbe à l'infinitif, noms à leur forme singulier, …)
  - Racinisation (stemming) : supprimer suffixes et flexions (pour l'anglais : retrieve, retrieving, retrieval, retrieved, retrieves ⇨ retriev)
  - Troncature à X caractères
  - Utilisation des n-grammes (caractères) ou (wordpiece)

Normalisation par Lemmatisation

- Réduire les variantes flexionnelles des mots à leur forme de base
- Ex.
  - *am, are, is* → *be*
  - *car, cars, car's, cars'* → *car*
  - *chante, chantons, chanterons* → *chanter*
- Utilisation d'un lexique (dictionnaire)

- Analyse grammaticale fine
  - Tree-tagger (http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/)

Normalisation par racinisation

- Utilisation de règles de transformations
  - règle de type : condition action
    - Ex : si mot se termine par s supprimer la terminaison
    - Plusieurs algos les plus connus : Porter, Lovins
    - Stemmer Snowball (http://snowball.tartarus.org/) disponibles en téléchargement

Ex. de résultats d'une normalisation avec Porter

- Texte original:

  Text representation is one of the fundamental problems in text mining and Information Retrieval (IR).

  It aims to numerically represent the unstructured text documents to make them mathematically computable"

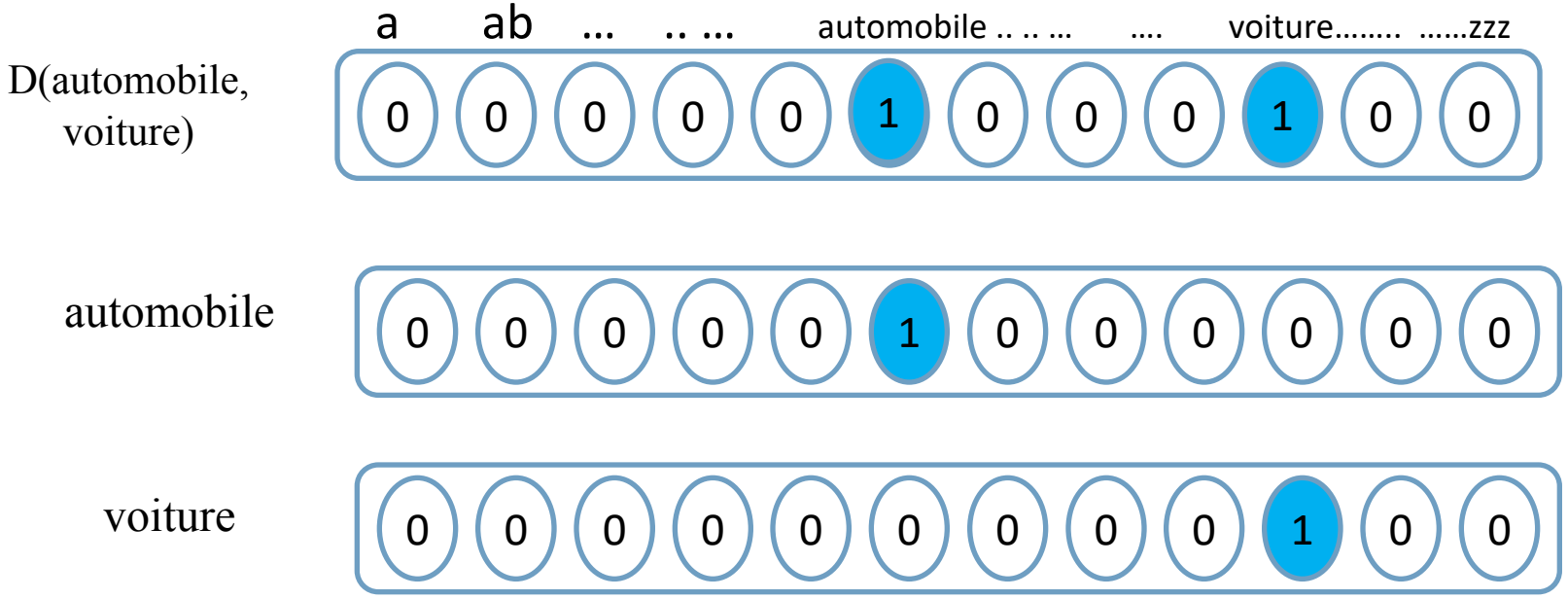
- Texte après porter + suppression mots vides

  'text': 3, 'represent': 1, 'one': 1, 'fundament': 1, 'problem': 1, 'mine': 1, 'inform': 1, 'retriev': 1, '(': 1, 'ir': 1, ')': 1, '.it': 1, 'aim': 1, 'numer': 1, 'repres': 1, 'unstructur': 1, 'document': 1, 'make': 1, 'mathemat': 1, 'comput': 1

Le vocabulaire final dépend du tokenizer (stemmer/lemmatiseur)

Représentation en “sac de mots” (Bag of Words)

- Représentation formelle
  - Ensembliste
    - D(automobile, voiture, ..)
  - Vectorielle (“One-hot representation (local)”)
    - D(automobile, voiture)



Vocabulaire = tous les mots du de la collection de textes

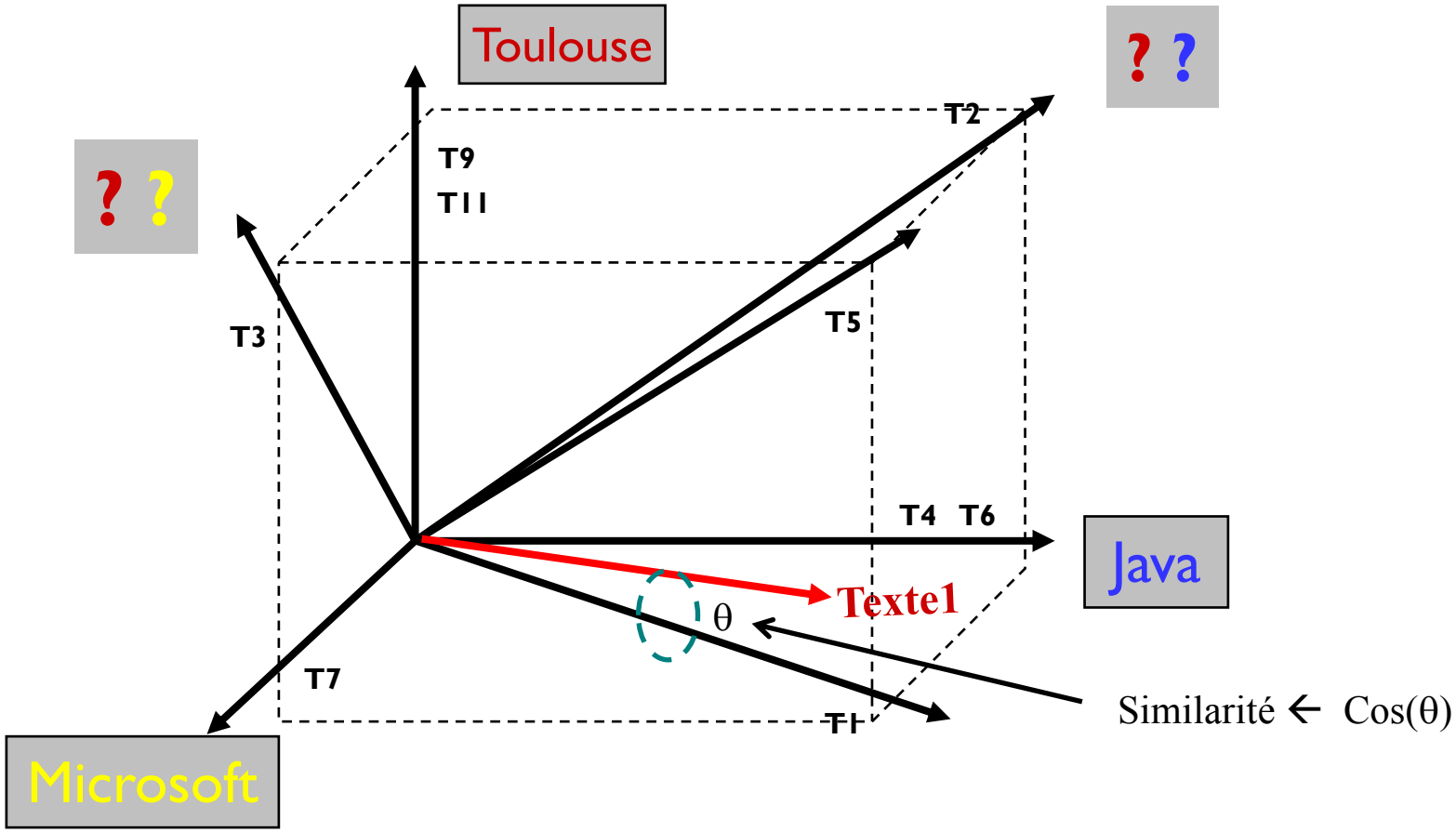
Representation vectorielle

- Une collection de *n* documents (Textes) et *M* termes distincts peut être représentée sous forme de matrice

$$
\begin{array}{c|cccc}
 & t_1 & t_2 & \ldots & t_M \\
D_1 & w_{11} & w_{21} & \cdots & w_{M1} \\
D_2 & w_{12} & w_{22} & \cdots & w_{M2} \\
\vdots & \vdots & \vdots & & \vdots \\
\vdots & \vdots & \vdots & & \vdots \\
D_n & w_{1n} & w_{2n} & \cdots & w_{Mn}
\end{array}
$$

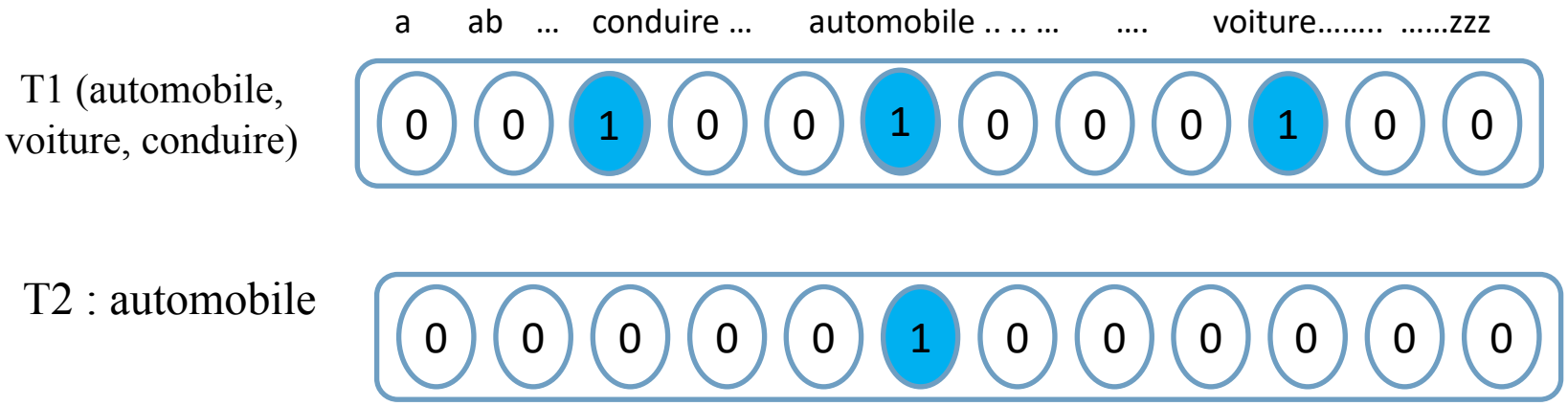
Représentation vectorielle

- Similarité vectorielle (Text matching - appariement de textes).



Représentation vectorielle

- Représentation vectorielle permet de calculer facilement des similarités entre deux textes.



Similarité (T1, T2), on peut utiliser toutes les similarités vectorielles

- Inner (T1,T2)=1 → $\sum x_i * y_i$

- Cosinus (T1, T2) = $\frac{1}{\sqrt{(1*3)}} = \frac{1}{\sqrt{3}}$ → $\frac{\sum x_i * y_i}{\sqrt{\sum x_i^2 * \sum y_j^2}}$

Importance des mots : aller au delà d'un simple comptage

Luhn's idea (1958): automatic indexing based on statistical analysis of text

Hans Peter Luhn (IBM)

"It is here proposed that the frequency of word occurrence in an article furnishes a useful measurement of word significance. It is further proposed that the relative position within a sentence of words having given values of significance furnish a useful measurement for determining the significance of sentences. The significance factor of a sentence will therefore be based on a combination of these two measurements. " (Luhn 58)

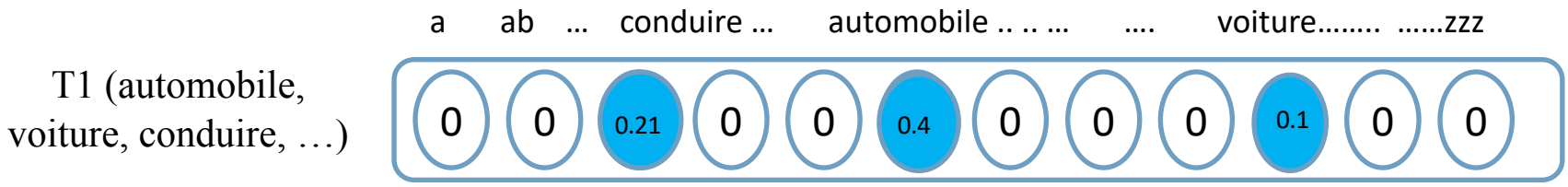
LUHN, H.P., 'A statistical approach to mechanised encoding and searching of library information', *IBM Journal of Research and Development,* **1**, 309-317 (1957).
LUHN, H.P., 'The automatic creation of literature abstracts', *IBM Journal of Research and Development*, **2**, 159-165 (1958).

Pondération : *tf.Idf*

- Deux facteurs *tf.idf*
  - *tf* : Fréquence du terme dans le document
  - *Idf* : fréquence (inverse) du terme dans la collection

$$tf = \begin{cases} freq(t,d) \\ 1+\log(freq(t,d)) \\ \dfrac{freq(t,d)}{\max_{\forall t' \in d}(t',d)} \\ \dfrac{freq(t,d)}{\sum_{\forall t' \in d} freq(t',d)} \end{cases} \qquad idf(t) = \begin{cases} \log\left(\dfrac{N}{n_t}\right) \\ \log(\dfrac{N-n_t}{n_t}) \end{cases}$$

- Représentation pondérée d'un texte :



Fonctions de ponderation : tf.idf

- PIV (vector space model)

$$\frac{1+\ln(1+\ln(c(w,d)))}{(1-s)+s\frac{|d|}{avdl}}\cdot c(w,q)\cdot\ln\frac{N+1}{df(w)}$$

TF

IDF

- DIR (language modeling approach)

$$c(w,q)\times\ln\left(1+\frac{c(w,d)}{\mu\cdot p(w|C)}\right)+|q|\cdot\ln\frac{\mu}{\mu+|d|}$$

Length Norm.

- BM25 (classic probabilistic model)

$$\ln\frac{N-df(w)+0.5}{df(w)+0.5}\cdot\frac{(k_1+1)\times c(w,d)}{k_1((1-b)+b\frac{|d|}{avdl})+c(w,d)}\cdot\frac{(k_3+1)\times c(w,q)}{k_3+c(w,q)}$$

- PL2 (divergence from randomness)

$$c(w,q)\cdot\frac{tfn_w^d\cdot\log_2(tfn_w^d\cdot\lambda_w)+\log_2 e\cdot(\frac{1}{\lambda_w}-tfn_w^d)+0.5\cdot\log_2(2\pi\cdot tfn_w^d)}{tfn_w^d+1}$$

$$tfn_w^d=c(w,d)\cdot\log_2(1+c\cdot\frac{avdl}{|d|}),\lambda_w=\frac{N}{c(w,C)}$$

Retour sur la Segmentation (Tokenization) : Est-il necessaire d'identifier des mots ?

- Différentes approches
  - basées sur les espaces/ponctuation
    - "Global warming is a long-term (rise) in the average temperature of the Earth's"

      ['global', 'warm', 'long-term', '(', 'rise', ')', 'averag', 'temperatur', 'earth', "'s"]
  - basées sur n-grams (exemple bigrams) (prendre les mots 2 à 2 avec overlap)
    - Bi-grams [('global', 'warm'), ('warm', 'long-term'), ('long-term', '('), ('(', 'rise'), ('rise', ')'), (')', 'averag'), ('averag', 'temperatur'), ('temperatur', 'earth'), ('earth', "'s")]
  - basées sur les sous-mots (WORDPIECE) (popularisé par BERT)
    - Les mots fréquents vont rester tels quels, les mots peu fréquents seront subdivisés en sous mots.
    - "I", "like", "natural", "lang", "##uage", "process", "##ing"

Le terme utilisé en anglais pour désigner ces mots, ngrams, sous-mots : un « token »

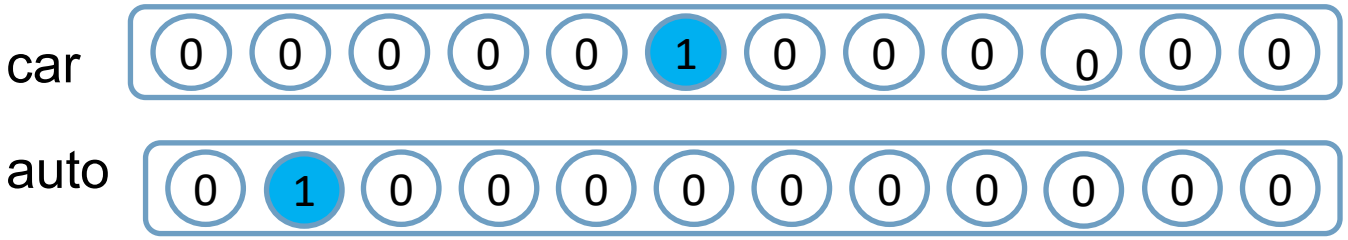
Exemple pratique sur Python

Utilisation des bibliothèques :

- NLTK
- Spacy
- Comparaison des Tokenizers utilisés dans les LLM tels que GPT

Limites de la représentation en sac de mots (1)

- 1- Les mots synonymes ont des vecteurs différents :
  - « Car » et « auto »



- 2- Les mots polysémiques ont les mêmes vecteurs :
  - « salsa (danse) », « salsa (sauce) ».

Limites de la représentation en sac de mots (2)

- 3- L'ordre des mots n'est pas pris en compte (la représentation est hors-contexte)

  - Texte 1 : Un garçon mange une pomme
  - Texte 2 : Une pomme mange un un garçon

	a	ab	...	.. ...	garçon	mange…….. pomme…	…
Texte 1	0	0	0	0	0	1	0	1	0	1	0	0
Texte 2	0	0	0	0	0	1	0	1	0	1	0	0

Comment résoudre ces problèmes ?

- Quels modèles pour tenir compte de l'ordre des mots ?→ Chapitre 2
  - Modèle de langue/language Model
    - Modèles probabilistes permettant de mieux prendre en compte la séquentialité des mots mieux rendre compte de la manière dont les mots sont agencés

- Quels modèles pour représenter le sens ? → Chapitre 3
  - Bases de connaissances/ontologies
    - WordNet, …
  - Modèles computationnels
    - Word2Vec (modèles statiques)
    - BERT, GPT(modèles contextualisés) …