Cours de Traitement Automatique du Langage

Chapitre 02 : Traitements de base

DI5 – 2021-22

Polytech ’Tours J.Y. RAMEL

Introduction

- Un texte est structuré en paragraphes, phrases, mots et caractères
- Certaines taches nécessite d'exploiter cette structure
- Notamment, la plupart des taches se basent sur les mots comme unité de traitement

- Au préalable, il est nécessaire d'exécuter des pré-traitements
  - Filtrer les données non nécessaires
  - Normaliser les mots pour limiter les variations à traiter
  - Segmenter le texte en mots ou phrases
  - Analyse morphologique (formation des mots, réductions des formes)

- Des traitements et outils basiques permettent d'exécuter ces procédures :
  - Utilisation d'expressions régulières, substitution, …
  - Calcul de distance d'édition

Expressions régulières

- Utilisation d'E.R. pour chercher des chaines de caractères dans un texte
  - Reconnaissance des langages de type 3 (langages réguliers dans la hiérarchie de Chomsky)
  - Utiles pour la normalisation, l'analyse lexicale (séparation des mots), la lémmatisation
  - Encore beaucoup utilisée pour l'analyse / la sélection de texte

- Exemple

ER	Sens	Exemple
.	n'importe quel caractère	***beg.n*** : I begun at the beginning.
[aeuio]	caractères spécifiques	***[Ll][ae]*** : Le chat mange la sourie.
[a-e]	plage de caractères	***[A-Z]..*** : J'ai vu Karim.
[^aeuio]	exclure des caractères	***[^A-Z]a.*** : J'ai vu Karim.
c?	un ou zéro	***colou?r*** : It is colour or color.
c*	zéro ou plus	***No*n*** : Nn! Non! Noooooooon!
c+	un ou plus	***No+n*** : Nn! Non! Noooooooon!
c{n}	n occurrences	***No{3}n*** : Nn! Non! Noon! Nooon!
c{n,m}	de n à m occurrences	***No{1,2}n*** : Nn! Non! Noon! Nooon!
c{n,}	au moins n occurrences	***No{2,}n*** : Nn! Non! Noon! Nooon!
c{,m}	au plus m occurrences	***No{,2}n*** : Nn! Non! Noon! Nooon!

Expressions régulières

- Pas facile de prévoir tous les cas ➔ ER complexes

- Exemple : Trouver toutes les instaces du mot “the” dans un texte
  - ER = the
  - Faux négatif → On loupe toutes les instances avec majuscule
  - ER = [tT]he
  - Faux positif → “other”, “theology”, …
  - ER = [^a-zA-Z][tT]he[^a-zA-Z]

- Les expressions régulières jouent encore un rôle étonnamment important
  - Les séquences sophistiquées d'expressions régulières sont souvent le premier modèle pour tout traitement de texte
  - Plus qu’uniquement la detection, les ER permettent la substitution de texte (simplifcation, lemmatisation, …)
  - Cf commandes unix, python, …

Niveau caractères – Distance d'edition

- Distance d'Edition: utilisée pour mesurer la différence entre deux chaınes de caractères
  - Correcteurs orthographiques
  - Utile pour la recherche approximative

- Operations d'édition
  - Insertion: insertion d'un caractère dans une chaıne
  - Suppression: suppression d'un caractère d'une
  - Substitution: substitution d'un caractère par un autre
  - Transposition: changement de l'ordre de deux caractères

- Distance
  - de Hamming: permet seulement la substitution
  - Distance de Levenshtein: permet l'insertion, la suppression et la substitution
  - Distance de Jaro: permet la transposition
  - Distance de Damerau–Levenshtein: permet l'insertion, la suppression, la substitution et la transposition entre deux caracteres adjacents

Niveau caractères – Distance d'edition

Algorithme

- Pour calculer D[n,m], on utilise la programmation dynamique
- X ; une chaine source de longueur n - Y : chaine destinataire de longueur m
- D[i, j] est la distance d'édition entre les sous-chaines X[1..i] et Y[1..j] - D[0,0] =0

$$D[i,j] = \min \begin{cases} D[i-1,j] + 1 \text{ //Suppression} \\ D[i,j-1] + 1 \text{ //Insertion} \\ D[i-1,j-1] + \begin{cases} 2 & \text{si } X[i] \neq Y[j] \\ 0 & \text{sinon} \end{cases} \end{cases}$$

	#	e	x	e	c	u	t	i	o	n
#	0	← 1	← 2	← 3	← 4	← 5	← 6	← 7	← 8	← 9
i	↑ **1**	↖←↑ 2	↖←↑ 3	↖←↑ 4	↖←↑ 5	↖←↑ 6	↖←↑ 7	↖ 6	← 7	← 8
n	↑ 2	↖←↑ **3**	↖←↑ 4	↖←↑ 5	↖←↑ 6	↖←↑ 7	↖←↑ 8	↑ 7	↖←↑ 8	↖ 7
t	↑ 3	↖←↑ 4	↖←↑ **5**	↖←↑ 6	↖←↑ 7	↖←↑ 8	↖ 7	←↑ 8	↖←↑ 9	↑ 8
e	↑ 4	↖ 3	← 4	↖← **5**	← **6**	← 7	←↑ 8	↖←↑ 9	↖←↑ 10	↑ 9
n	↑ 5	↑ 4	↖←↑ 5	↖←↑ 6	↖←↑ 7	↖←↑ **8**	↖←↑ 9	↖←↑ 10	↖←↑ 11	↖↑ 10
t	↑ 6	↑ 5	↖←↑ 6	↖←↑ 7	↖←↑ 8	↖←↑ 9	↖ **8**	← 9	← 10	←↑ 11
i	↑ 7	↑ 6	↖←↑ 7	↖←↑ 8	↖←↑ 9	↖←↑ 10	↑ 9	↖ **8**	← 9	← 10
o	↑ 8	↑ 7	↖←↑ 8	↖←↑ 9	↖←↑ 10	↖←↑ 11	↑ 10	↑ 9	↖ **8**	← 9
n	↑ 9	↑ 8	↖←↑ 9	↖←↑ 10	↖←↑ 11	↖←↑ 12	↑ 11	↑ 10	↑ 9	↖ **8**

Segmentation du texte

Segmentation en paragraphes, mots, expressions particulières

- Tokenisaton➔ Détection et dénombrement des mots (ou parfois N-grams)
  - Parfois simple → ponctuation – Parfois impossible selon la langue
- Un nouveau paragraphe est marqué par au moins un retour à la ligne (<p> en HTML).
- Une fin de phrase est marquée par un point (ou une autre marque de ponctuation) Pas toujours → Le point n'est pas seulement utilisé pour marquer une phrase
- La casse: les phrases commencent par un majuscule → pas toujours
- Un mot est délimité par une espace → Pas toujours
- Catégorie grammaticale : les catégories des mots avant le point peuvent aider la décision
- Longueur du mot: les abréviations sont moins longues
- Noms propres: les noms propres commencent par un majuscule ; ils peuvent ne pas être le début d'une phrase
- Les expressions avec des mots multiples : les expressions numériques comme les dates

Question : une ER simple pour délimiter les phrases ?

Segmentation du texte

Segmentation en paragraphes, mots, expressions particulières

- Tokenisaton➔ Détection et dénombrement des mots (ou parfois N-grams)
  - Parfois simple → ponctuation – Parfois impossible selon la langue
- Un nouveau paragraphe est marqué par au moins un retour à la ligne (<p> en HTML).
- Une fin de phrase est marquée par un point (ou une autre marque de ponctuation) Pas toujours → Le point n'est pas seulement utilisé pour marquer une phrase
- La casse: les phrases commencent par un majuscule → pas toujours
- Un mot est délimité par une espace → Pas toujours
- Catégorie grammaticale : les catégories des mots avant le point peuvent aider la décision
- Longueur du mot: les abréviations sont moins longues
- Noms propres: les noms propres commencent par un majuscule ; ils peuvent ne pas être le début d'une phrase
- Les expressions avec des mots multiples : les expressions numériques comme les dates

Question : une ER simple pour délimiter les phrases ? → **/[.?!]/** et les mots ? **/[ ]+/**

Normalisation et filtrage du texte

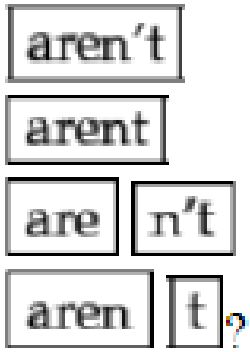
Normalisation du texte

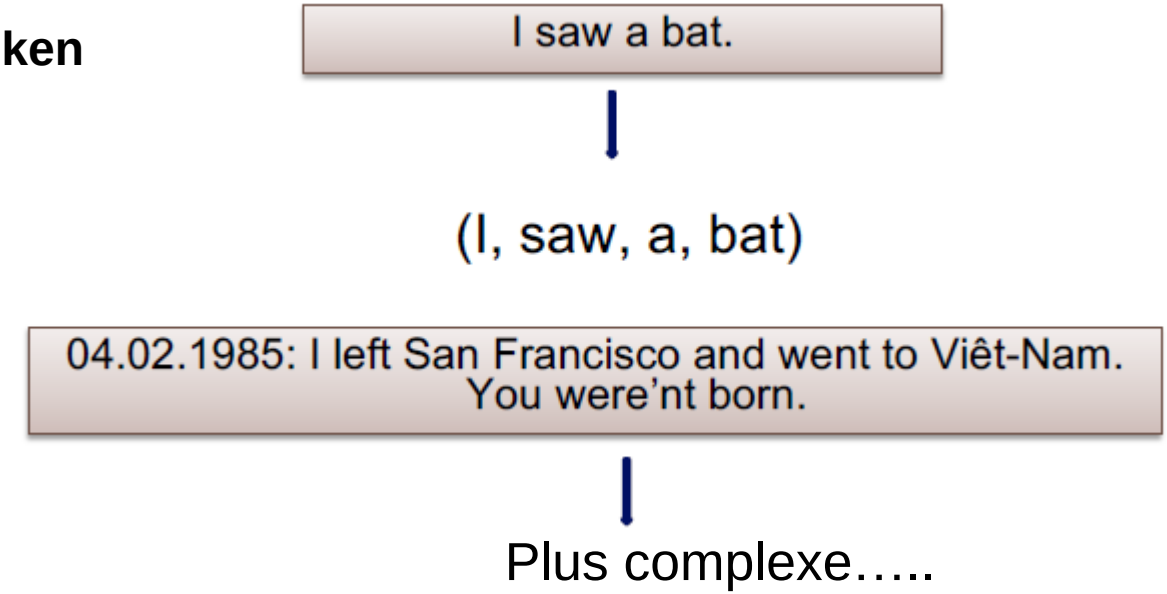
PROBLEMATIQUE:

Un texte peut contenir des variations du même terme (lemmatisation). Aussi, dans des taches comme la recherche d'information, on n'a pas besoin d'avoir le contenu exacte du texte.

SOLUTION: transformer le texte a une forme canonique

Tokenization ➔ découpage en token





Normalisation et filtrage du texte

Tokenization ➔ découpage en token

Méthodes

- Heuristique-based :
  - Utilisation d'un grand vocabulaire → Utilisation de lexiques (dictionnaires)
  - Prendre la correspondance la plus longue du vocabulaire
  - Ex : Je suis allé à San Francisco -> (je, suis allé, San Francisco)
  - Utilisation d'une heuristique pour les mots inconnus
- Apprentissage automatique
  - entraînés sur des mots segmentés à la main !!!
  - Modèles de Markov cachés, champs aléatoires conditionnels,…
  - Réseaux neuronaux (récurrents)

Challenges

  - https://www.kaggle.com/c/text-normalization-challenge-english-language

Normalisation et filtrage du texte

Filtrage du texte

PROBLEMATIQUE: le texte peut contenir des caractères, des mots et des expressions qui peuvent entraver son traitement

SOLUTION: suppression

Pourquoi ?

- Les caractères spéciaux comme les caractères non imprimables peuvent mener a des traitements erronés
- Les Hapax : termes isolés
- Les mots vides: les mots non significatifs comme les prépositions, articles et les pronoms.
- en anglais : stop words
  - dans la recherche d'information, il est inutile de les indexer

Normalisation et filtrage du texte

Normalisation du texte

- Acronymes et les abréviations
  - forme standard : US→USA, U.S.A.→USA
  - version longue : M.→ Monsieur
- Formater les valeurs comme les dates et les nombres de la même facon
  - Conversion a la forme textuelle : 1205 DZD → Mille deux cents cinq dinarsalg´eriens
  - Format spécifique : 12 Janvier 1986, 12.01.86 → 1986-01-12
  - Remplacement par le type : 12 Janvier 1986→DATE, kariminfo0@gmail.com → EMAIL
- Transformer les majuscules en minuscules.
  - Texte→texte
  - Des fois, il faut laisser la casse telle qu'elle est, comme dans les noms propres (Will)
- Contractions
  - y'll→you all, s'il→si il
- Diacritiques
  - desaccentuation : systeme→système
  - dé-vocalisation : يَدرُسُ → يدرس.
- Encodage
  - il faut utiliser le même encodage supporté dans le traitement

Normalisation et filtrage du texte

Exemple de normalisation du texte

Input : « I like this actor but I am not convinced by his play »

Tokenization : (I, like, this, actor, but, I, am, not, convinced, by, his, play)

Filtering: (~~I~~, like, ~~this,~~ actor, ~~but,~~ I, am, not, convinced, ~~by,~~ ~~his,~~ play)

Morphologie - Stemming

Réduction de formes (**racinisation - stemming**)

- Suppression des affixes → Résultat radical (racine / stem)
- Exemple : Anglais : stemming → stem – Français : chercher → cherch
- Base de données : stocker tous les termes et leurs racines dans une table
- Statistiques: utiliser un modèle de langue (N-Gram) pour estimer la position de troncation

Algorithme de Porter [Porter et al., 1980]

- Un ensemble de règles condition/action associé à un framework pour créer des « racinateurs »
- Dans le cas de Porter, analyse des suffixe et condition sur l'affixe
- Condition sur la racine, la longueur, la fin, si elle contient des voyelles, etc.

$$\begin{aligned} \text{ATIONAL} &\rightarrow \text{ATE} \quad (\text{e.g., relational} \rightarrow \text{relate}) \\ \text{ING} &\rightarrow \epsilon \quad \text{if stem contains vowel (e.g., motoring} \rightarrow \text{motor)} \\ \text{SSES} &\rightarrow \text{SS} \quad (\text{e.g., grasses} \rightarrow \text{grass}) \end{aligned}$$

Morphologie - Stemming

Exemples stemming – resultats variables…

Sample text: Such an analysis can reveal features that are not easily visible from the variations in the individual genes and can lead to a picture of expression that is more biologically transparent and accessible to interpretation

Lovins stemmer: such an analys can reve featur that ar not eas vis from th vari in th individu gen and can lead to a pictur of expres that is mor biolog transpar and acces to interpres

Porter stemmer: such an analysi can reveal featur that ar not easili visibl from the variat in the individu gene and can lead to a pictur of express that is more biolog transpar and access to interpret

Paice stemmer: such an analys can rev feat that are not easy vis from the vary in the individ gen and can lead to a pict of express that is mor biolog transp and access to interpret

Morphologie - Lemmatisation

Des langues permettent la formation des mots en utilisant la flexion (ex. conjugaison) et la dérivation (ex. nominalisation)

- Flexion: variation morphologique d'un mot selon les traits grammaticaux (nombre, genre, etc.)
  - Conjugaison des verbes
  - Déclinaison des noms, pronoms, adjectifs et déterminants
- Dérivation: variation morpho. d'un mot pour créer un nouveau lexème ou changer de catégorie
  - Création un nouveau lexème : couper → découper,...
  - Changement de catégorie : classer→ classement, classeur ; Adjectif (fatiguer → fatigant)

Gestion des conjugaisons

- Base de données
  - stocker la conjugaison des verbes dans une base de données
- Modèles (template)
  - stocker la conjugaison de certains verbes modèles et utiliser une liste pour indexer les verbes similaires
- Règles : utilisation des règles SI-SINON et des expressions régulières
  - Exemple, conjugaison des verbes en arabe, anglais, français et japonais
  - https://github.com/kariminf/jslingua

Morphologie - Lemmatisation

Réduction de formes (lemmatisation)

- Chercher la forme canonique d'un mot (anglais : lemmatization)
- Résultat : lemme (anglais : lemma)
  - Exemple, comprennent→ comprendre, better→ good

- Nécessite d'avoir le contexte du mot
  - Ex. saw→(V) see ou (N) saw

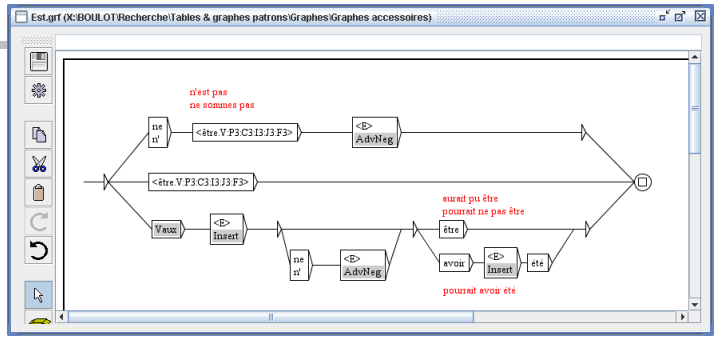
- Nécessite des bases lexicales
  - https://www.nltk.org/api/nltk.stem.htm (NLTKWordNetLemmatizer)
  - https://github.com/sloria/textblobhttps

- Apprentissage automatique
  - https://opennlp.apache.org/

Lemmatisation "morphy" de Wordnet

Data : mot, catégorie
Result : liste des lemmes possibles
si *mot* ∈ *list_exceptions[catégorie]* **alors**
  return chercher_dans_dictionnaire({mot} ∪ list_exceptions[catégorie]);
fin
formes = {mot}
tant que *formes* ≠ ∅ **faire**
  formes = supprimer_les_affixes(formes, catégorie);
  résultats = chercher_dans_dictionnaire({mot} ∪ formes);
  si *résultats* ≠ ∅ **alors**
    return résultats ;
  fin
fin
return ∅;

Segmentation du texte



Séparation, détection de mots, phrases ou expressions
Les approches (CF TP Unitex)

- Par règles : en utilisant des expressions régulières
  - https://tln.lifat.univ-tours.fr/version-francaise/ressources/tutoriels-unitex
  - https://www.nltk.org/api/nltk.tokenize.html
  - https://nlp.stanford.edu/software/tokenizer.shtmlhttps://spacy.io/
  - https://github.com/kariminf/jslingua
  - https://github.com/linuxscout/pyarabic

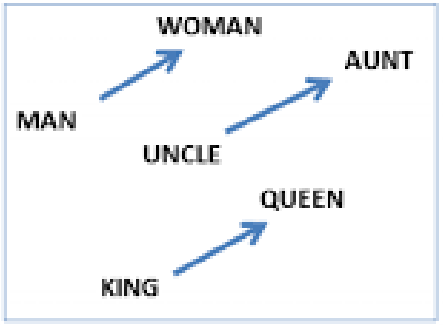
- Statistique : en utilisant un modele de langue pour calculer la probabilité qu'un caractère/une expression marque la fin d'un mot ou d'une phrase
  - https://nlp.stanford.edu/software/segmenter.htmlhttps://opennlp.apache.org/

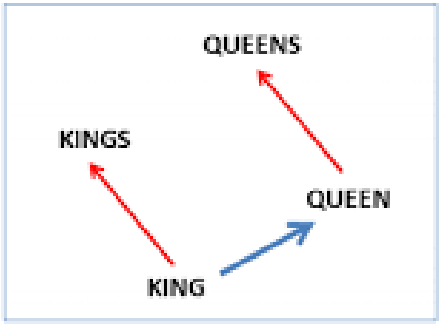
Représentation de texte ou sequences de mots

- Analyse fréquentielle
  - Bag of Words

	call	time	date	conference	release	meeting	corporation	earnings
document 1	2	1	3	2	1	1	1	
document 2	1		2	1	2	1	1	1
document 5		1	2		2	1	1	1
document 6	1	2	1	1	3	1	1	1
document 7	1						1	
document 8			1		1		1	1
document 9	2		1	3	1	1	1	1
document 10	2	1		1	1		1	1
document 13					1			2
document 14							3	
document 15	1			2			1	2

- Word embedding



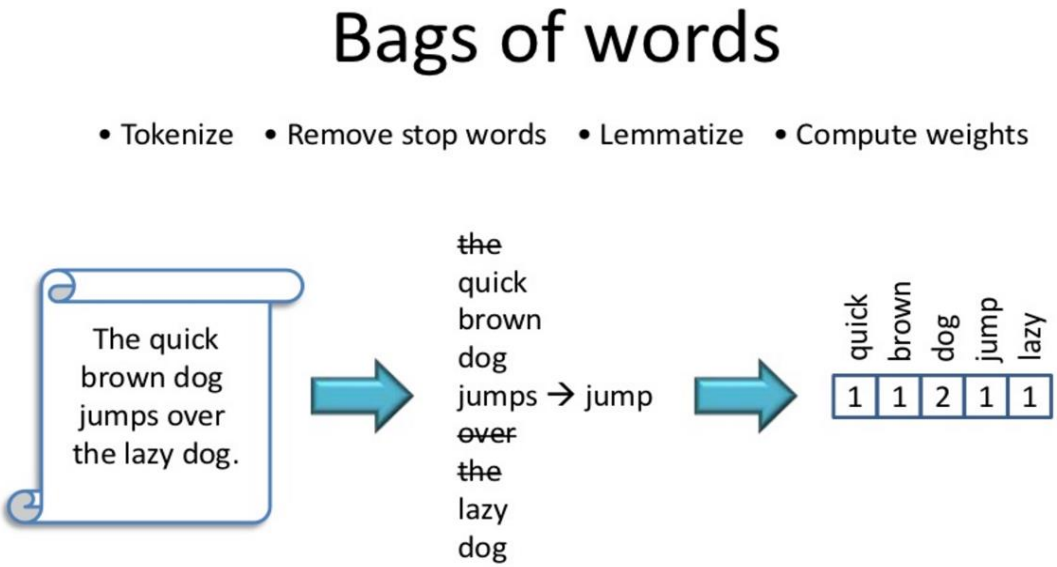


(Mikolov et al., NAACL HLT, 2013)

Représentation de texte ou sequences de mots

Représentation des sacs de mots (BOW)

- 1 document = 1 vecteur (a_1, ..., a_N)
- a_i = nombre d'occurrences du mot w_i dans le document d.
- Tokenization souvent nécessaire



	call	time	date	conference	release	meeting	corporation	earnings
document 1	2	1	3	2	1	1	1	
document 2	1		2	1	2	1	1	1
document 5		1	2		2	1	1	1
document 6	1	2	1	1	3	1	1	1
document 7	1						1	
document 8			1		1		1	1
document 9	2		1	3	1	1	1	1
document 10	2	1		1	1		1	1
document 13					1			2
document 14							3	
document 15	1			2			1	2

Matrice document-termes

Représentation de texte ou sequences de mots

Représentation basée sur TF-IDF

(Term Frequency – Inverse Document Frequency)

- mesure statistique utilisée pour évaluer la représentativité d'un mot pour un document particulier dans une collection de documents

- 1 document = 1 vecteur (a_1... a_N)
  a_i = TF-IDF du mot w_i dans le document d

$$TFIDF(w,d) = TF_{w,d} . IDF_{w,d}$$
$$= TF_{w,d} . \left(\left(\log_2 \frac{M}{DF_w} \right) \right)$$

M : number of documents
TF : Term Frequency
Number of occurrences of w in d.
Or boolean: tf(w,d) = 1 if w in d, 0 otherwise
DF : Document Frequency
Number of documents with the word w
This value grows proportionally to the occurrences of the word in the document (TF) but its effect is countered by the occurrences of the word in every other document (IDF)

Représentation de texte ou sequences de mots

Exercice :
Calculer le TF-IDF du mot "director" pour le document d

- TF-IDF(" directeur ", d) = ?
- La base de données contient 1000 documents
- Le document d contient 3 fois le mot "director".

A. 70 textes contiennent le mot "director", le mot "director" apparaît 134 fois dans la base de données

B. 900 textes contiennent le mot "director". Le mot "director" apparaît 1014 fois dans la base de données

$$TFIDF(w,d) = TF_{w,d}.IDF_{w,d}$$
$$= TF_{w,d}.\left(\left(\log_2 \frac{M}{DF_w}\right)\right)$$

M : number of documents
TF : Term Frequency
Number of occurrences of w in d.
Or boolean: tf(w,d) = 1 if w in d, 0 otherwise
DF : Document Frequency
Number of documents with the word w

Représentation de texte ou sequences de mots

Exercice :
Calculer le TF-IDF du mot "director" pour le document d

- TF-IDF(" directeur ", d) = ?
- La base de données contient 1000 documents
- Le document d contient 3 fois le mot "director".

A. 70 textes contiennent le mot "director", le mot "director" apparaît 134 fois dans la base de données

B. 900 textes contiennent le mot "director". Le mot "director" apparaît 1014 fois dans la base de données

$$3.\left(\log_2 \frac{1000}{70}\right) = 11{,}5$$

$$3.\left(\log_2 \frac{1000}{900}\right) = 0.45$$

Représentation de texte ou séquences de mots

Mesure de similarité entre documents ou séquences

Similitude en cosinus

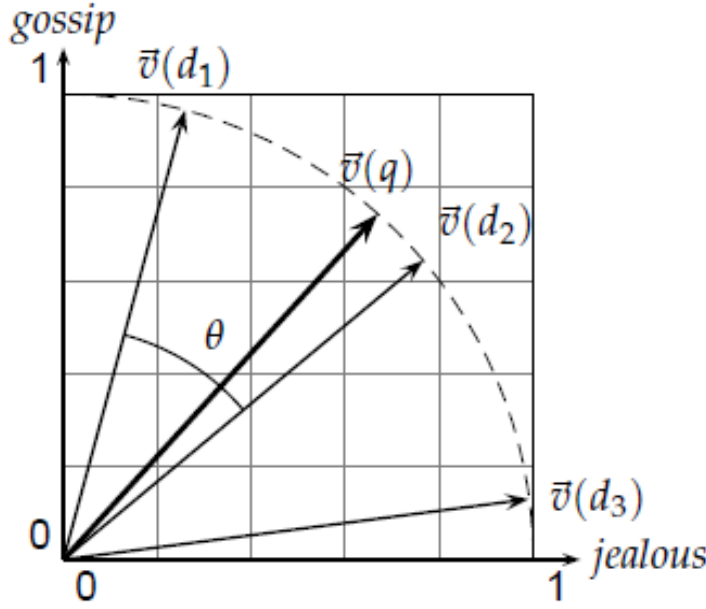
- Souvent utilisée pour mesurer le lien entre 2 vecteurs de documents d_1 et d_2

$$\text{sim}(d_1, d_2) = \frac{\vec{V}(d_1) \cdot \vec{V}(d_2)}{|\vec{V}(d_1)||\vec{V}(d_2)|},$$

Inconvénients des BOW

- Fonctionne que sur des documents conséquents
- Invariant à l'ordre des termes dans le document

Ex: These two sentences are represented by the same vector
"Mary is quicker than John"
"John is quicker than Mary"



Bibliographie

Bibliographie

- Indurkhya, N. and Damerau, F. J. (2010).Handbook of Natural Language Processing.Chapman & Hall/CRC, 2nd edition.
- Jurafsky, D. and Martin, J. H. (2019).Speech and Language Processing. https://web.stanford.edu/˜jurafsky/slp3/
- Porter, M. F. et al. (1980).An algorithm for suffix stripping.Program, 14(3):130–137