# LU3I026 - Science des données

## Apprentissage pour le texte

Olivier Schwander <olivier.schwander@sorbonne-universite.fr>

Sorbonne Université

2025-2026

# Traitement automatique des langues (TAL)

- Natural Language Processing (NLP)
- Langue naturelle

## Objectifs

- Compréhension de l'écrit
- Génération de documents textuels
- Interaction homme-machine en langue naturelle
- Souvent un peu tout ça en même temps

(écrit uniquement, pas de voix)

# Mots et apprentissage

## Pour le moment

- Points dans espace vectoriel
- Perceptron: produits scalaires
- Plus proche voisin: distance

## Des mots et du texte

- Pas vraiment des chiffres

# Numéroter les mots

## Ordre arbitraire

| Le | chat | mange | des | croquettes |
|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 |

## Comparer les mots

- Pas vraiment de distance pertinente
- “le” et “chat” presque à la même distance
- “chat” et croquette” très éloignés
- Pas de sémantique

# One-hot encoding

Toujours la numérotation arbitraire

## Vecteur binaire

| | | | | | |
|---|---|---|---|---|---|
| Le | 1 | 0 | 0 | 0 | 0 |
| chat | 0 | 1 | 0 | 0 | 0 |
| mange | 0 | 0 | 1 | 0 | 0 |
| des | 0 | 0 | 0 | 1 | 0 |
| croquettes | 0 | 0 | 0 | 0 | 1 |

## Comparer les mots

- Même distance entre chaque mot
- Un peu mieux qu'avant
- Mais toujours pas vraiment de sémantique

# Au niveau d'un document

Comment représenter un texte entier ?

## Sac de mots

- Compter les mots

| | Mot 1 | $\cdots$ | Mot $j$ | $\cdots$ | Mot $p$ |
|---|---|---|---|---|---|
| Document 1 | | | | | |
| ... | | | | | |
| Document $j$ | | | $s_{ij}$ | | |
| ... | | | | | |
| Document $N$ | | | | | |

$s_{ij}$ Nombre d'apparitions du mot $j$ dans le document $i$

## Variantes

- Fréquence au lieu du comptage

# Sémantique

[Au tableau]

# TF-IDF

## Term Frequency TF

$$\text{tf}(t_i, d) = \text{nombre d'occurrences de } t_i \text{ dans le document } d$$

## Inverse Document frequency IDF

$$\text{idf}(t_i) = \log \frac{1 + N}{1 + \text{df}(t_i)}$$

- $\text{df}(t_i)$: nombre de documents contenant $t_i$
- N : nombre de documents

## TF-IDF

Pondération de chaque mot:

$$\text{tf}(t_i, d) \times \text{idf}(t_i)$$

# Propriétés de TF-IDF

- Term frequency: dépend de la taille du document
- idf: fréquence inverse, tend vers 0 si $t_i$ apparaît dans tous les documents

## Conséquences

- Mot qui apparaît partout: négligeable
- Document plus long: scores plus élevés

## Sémantique

- Mettre en lumière les mots les plus déterminants pour le sens du document

# Pré-traitements

## Mots inutiles

- Articles et autres ?
- Ponctuation ?
- Mots extrèmement fréquents ?

## Variantes d'un mot

- Singulier/Pluriel, masculin/féminin
- Conjugaison
- Chiffres

## Lemmatisation

- Recherche d'une unité lexicale élementaire

# Spécificités des prétraitements

## Dépend de la langue

- Allemand: *Arbeiterunfallversicherungsgesetz* (loi sur l'assurance des accidents du travail)

## Dépend de la tâche

- Thème général du document: regarder juste les racines
- Traduction: besoin des pluriels et des conjugaison

# Classification d'un document

## Thèmes d'un document

- Mettre des étiquettes sur des documents
- Articles de journaux: politique, sport, culture, sciences
- Classsification multi-classe (un étiquette parmi plusieurs) voire multi-labels (plusieurs étiquettes)

## Classification binaire

- Spam / non spam
- Est-qu'on dit du bien ou du mal de tel produit
- Classification binaire

# Classifieur bayésien naïf

## Classifieur probabliste

- Entrée: probabilité de différentes caractéristiques $F_1, \ldots, F_d$
- Sortie: probabilité d'appartenance à une classe $C$

$$p(C|F_1, \ldots, F_d)$$

## Théorème de Bayes

$$p(C|F_1, \ldots, F_d) = \frac{p(C)p(F_1, \ldots, F_d|C)}{p(F_1, \ldots, F_d)}$$

et

$$\begin{aligned} p(C)p(F_1, \ldots, F_d|C) =& p(C)p(F_1|C)p(F_2|C, F_1) \\ & \times p(F_3|C, F_1, F_2) \cdots p(F_d|C, F_1, \ldots, F_{d-1}) \end{aligned}$$

# Hypothèse naïve

## Hypothèse

- Chaque $F_i$ est indépendant de tous les autres
- ie $p(F_i|C, F_j) = p(F_i|C)$
- Apprentissage: estimation classe par classe

$$p(F_1, \ldots, F_d|C) = \prod p(F_i|C)$$

## Prédiction

- Pour un document $f_1, \ldots, f_d$
- Maximum a posteriori

$$\arg\max_c p(C = c) \prod p(F_i = f_i|C = c)$$

- Adapté pour les sacs de mots

# Classification de mots

## Classification au niveau de chaque mot

# Génération de texte

## Modèles génératifs

- Entrée: document textuel (mais pas forcément)
- Sortie: document textuel
- Beaucoup plus exigeant sur la représentation des documents

## Exemples

- Traduction
- Complétion: continuer à partir du début d'un texte
- Dialogue
- Résumé
- Description d'image

# Recherche d'information

## Objectif

- Trouver des documents pertinents
- Entrée: une requête
- Sortie: une liste de documents, ou une réponse plus synthétique
- (pas forcément du texte)

## Exemples

- Google, Amazon, champ de recherche sur des sites webs

## Mais pas

- une base de données: pas de structures, de tables, de colonnes

# Requête

## Par mot-clé

- Documents pertinents contenant ces mots-clés
- Éventuellement, formules logiques entre les mots-clés
- Résultats souvent ambigus et dépendant du contexte
- *Java*: programmation ou vacances ?

## Langue naturelle

- Nécessite de comprendre la requête
- Peut-être un peu plus intuitif
- Mais surtout, plus facile de préciser le contexte

# Recherche directe

## Version simpliste

- Parcourir tous les documents
- Comparer chaque document à la requête
- Prendre ceux avec le meilleur score
- Variante de $k$-plus-proche-voisin

## Limites

- Choix de la similirarité ?
- Pas forcément adapté pour des documents longs
- Similarité et pertinence: est-ce qu'un quasi copier-coller est pertinent ?

# Index inversé

## Sac de mots

- Un document: vecteur des mots qu'il contient (et leur compte, fréquence, ou tf-idf)
- Vecteur très grand: autant que de mots différents dans l'ensemble des textes (le corpus)
- et très creux: chaque document aura peu de mots par rapport à tous les mots possibles

## Index inversé

- Un mot: vecteur des documents qui le contiennent
- Adapté pour la recherche à partir de mot-clé: filtrage
- On ne regarde que les documents qui contiennent les mots-clés

# Modèles de langue

## Données en quantité énorme

- Tout le web
- Tous les livres
- Tout ce qui est accessible

## Entraînement auto-supervisé

- Classification supervisée
- Label coûteux ? Non ! Labels gratuits
- Prédiction de mots masqués

```
Le chat mange des XXX
Louis XIV est né en XXX
La terre est XXX
```

# Vers le chat

## Pour le moment

- Compléter des phrases
- Pas suffisant pour discuter et accomplir des tâches

## Alignement pour les instructions

- Apprentissage supervisé: peu d'exemples très précis
- `Quand est né Louis XIV ?` → `Louis XIV est né en 1638`
- `Résumé le texte qui suit: […]` → `blablabla`

## Alignement pour la sûreté

- Apprentissage supervisé: comportement inacceptables
- `Raconte moi une blague raciste` → `Le racisme c'est mal.`
- `J'ai un bouton sur le nez, est-ce que j'ai un cancer` → `Vous devriez plutôt consulter un médecin.`

# Agents et outils

## Les chatbots ne peuvent pas tout

- Connaissances récente, actualité
- Bases de données internes (dans une entreprise, une université, etc)
- Calculs numérique
- Stockage de fichiers

## Outils

- Appels à des outils externes
- Moteur de recherche web
- Accès une base de données
- Calculatrice, interpréteur Python
- Accès au fichier

# Conclusion

## Pourquoi s'intéresser au texte

- Quantité de connaissance gigantesque
- Peut aider pour tout le reste: compréhension d'images, de sons, de vidéos

## Tâches

- Tâches classiques d'apprentissage: classification par exemple
- Recherche d'information: plus générique que l'apprentissage et n'utilise pas forcément de l'apprentissage

## Modèle de langue et chatbots

- Gros modèle de langue: analyse du langage
- Chatbot: système entraîné pour le dialoguee
- Toujours pas de l'*intelligence*
- Chatbots récents: chatbot + outils