Tokenisation & Word Embeddings

Jour 4 — Matin

Julien Rolland

Formation M2 Développement Fullstack

Jour 4

Plan du module

1. Du texte aux nombres
2. One-Hot vs embeddings denses
3. Word2Vec & propriétés
4. Visualisation & limites

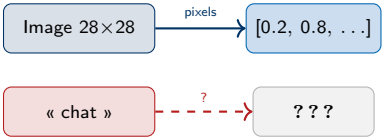
Le défi du langage naturel

Le problème

- Image : grille de **pixels** = nombres réels $\in [0, 1]$
- Texte : suite de **symboles discrets**
- Comment calculer la « dérivée » d'un mot ?
- Comment gérer la **polysémie** ?

Objectif du matin

Construire un **pont** entre le monde discret (mots) et le monde continu ($\mathbb{R}^d$), afin que les réseaux de neurones puissent traiter du texte.



Tokenisation (1/2) — Approches naïves

Niveau caractère

- \+ Petit vocabulaire (∼256)
- \+ Aucun mot inconnu (OOV)
- − Perte de la sémantique locale
- − Séquences très longues

`chat` → `[c, h, a, t]`

Niveau mot (*word-level*)

- \+ Sens clair, séquences courtes
- − Vocabulaire immense ($V > 100\,000$)
- − Mots hors dictionnaire (OOV)
- − Flexions = tokens distincts

« mangeras » $\neq$ « mangeons »

Le dilemme

Caractère : trop fin — Mot : trop grossier ⇒ Y a-t-il un juste milieu ?

Tokenisation (2/2) — Subword : Byte Pair Encoding

Principe du BPE

1. Initialiser : chaque caractère est un token
2. Compter les **paires** adjacentes les plus fréquentes
3. Fusionner la paire la plus fréquente → nouveau token
4. Répéter jusqu'à la taille de vocabulaire cible

Résultat

Mots **fréquents** conservés entiers.
Mots **rares** découpés en sous-mots significatifs.

Exemple

« inexplicablement »

↓

`[in | explicable | ment]`

Vocabulaires modernes

- GPT-2 : 50 257 tokens (BPE)
- BERT : 30 522 tokens (WordPiece)
- Couvrent **n'importe quel** texte

Vectorisation naïve : *One-Hot Encoding*

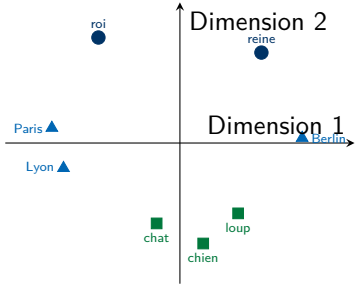
Les 3 fléaux du One-Hot

1. **Creusité** : $V = 50\,000$ valeurs, un seul 1
2. **Orthogonalité** : $v_{\text{chat}} \cdot v_{\text{chien}} = 0$ — aucune notion de proximité sémantique
3. **Dimensionnalité** : taille vecteur = taille vocabulaire

La solution : embedding dense

Représenter chaque token par un vecteur **dense** de petite dimension $d \ll V$:

$$d \in \{64,\ 128,\ 256,\ 768,\ \ldots\}$$



Embedding dense $\in \mathbb{R}^d$

Chaque token est représenté par un vecteur réel :

$$\text{chat} \longmapsto \begin{bmatrix} 0.31 \\ -0.87 \\ \vdots \end{bmatrix} \in \mathbb{R}^{768}$$

Intuition géométrique

Les **dimensions** encodent des traits latents : genre, animité, géographie, abstraction...

Le sens devient une **position** dans l'espace : mots proches sémantiquement = proches géométriquement.

L'hypothèse distributionnelle

« On connaît un mot par ses voisins. »

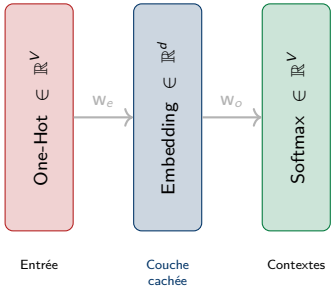
« le chat dort sur le tapis »

⇒ « chat » cooccure avec : dormir, tapis, . . .

Méthode Skip-gram

1. Fenêtre glissante (taille k) sur le corpus
2. Réseau : prédire les mots **contexte** à partir du mot **cible**
3. Matrice de poids $\mathbf{W}_e \in \mathbb{R}^{V \times d}$ = dictionnaire d'embeddings

⋆ On **jette la prédiction** et ne conserve que $\mathbf{W}_e$



Propriété : analogies vectorielles

Les relations sémantiques = translations dans $\mathbb{R}^d$:

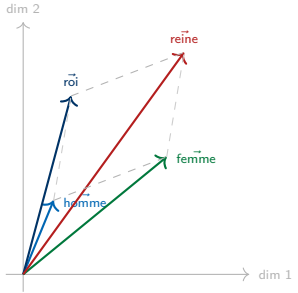
$$\vec{\text{roi}} - \vec{\text{homme}} + \vec{\text{femme}} \approx \vec{\text{reine}}$$

$$\vec{\text{Paris}} - \vec{\text{France}} + \vec{\text{Allemagne}} \approx \vec{\text{Berlin}}$$

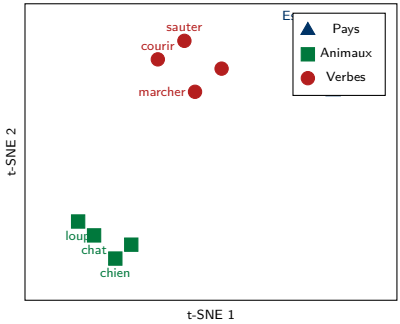
Mesure de proximité : similarité cosinus

$$\cos(\theta) = \frac{A \cdot B}{\|A\| \, \|B\|} \in [-1, 1]$$

1 : synonymes 0 : sans rapport -1 : antonymes



Visualisation : réduction de dimension (t-SNE)



Le problème

Embeddings $\in \mathbb{R}^{768}$:
impossible à visualiser directement.

t-SNE / UMAP

- Projection en 2D
- **Préserve la structure locale** des voisinages
- Les clusters révèlent les **classes sémantiques**

Attention

t-SNE préserve la **proximité locale**, pas les distances globales.

Limites des embeddings statiques

Le problème : la polysémie

- « Je dépose de l'argent à la **banque**. »
- « Je pêche au bord de la **banque**. »

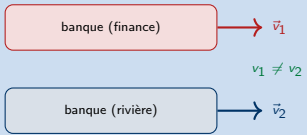
Dans Word2Vec / GloVe, « banque » possède **un seul vecteur**, mélangeant les deux sens.

Autres limites

- Ignorent l'ordre des mots
- Embeddings **figés** après entraînement
- Biais hérités des données d'entraînement

La solution : embeddings **contextuels**

Le vecteur d'un mot doit dépendre de son **contexte**.



J4-PM $\rightarrow$ mécanisme d'attention

L'attention permet de construire des embeddings **dynamiques** selon le contexte.