Master Informatique & Big Data

Fouille de données textuelles
Rapport approfondi sur Word2Vec : de la théorie à la pratique

2025/2026

Auteur : **Mevlut Cakin**

Table des matières

1 Introduction

Ce projet s'inscrit dans le cadre de l'analyse de données textuelles appliquée à une base de données de films. L'objectif est d'explorer l'utilisation des *word embeddings*, en particulier **Word2Vec**, pour extraire des représentations vectorielles porteuses de relations sémantiques à partir des synopsis, tags et métadonnées. Nous proposons une démarche complète : prétraitement, entraînement, visualisation et évaluation, ainsi qu'une interface **Dash** pour l'exploration interactive.

2 Contexte et objectifs

Nous visons à :

— Prétraiter un corpus de descriptions et métadonnées de films (tokenisation, normalisation, stopwords, détection de *phrases*).

— Entraîner un modèle **Word2Vec** (*skip-gram*) adapté au corpus.

— Évaluer la qualité des embeddings (distribution des similarités, cohérence *genres vs. voisins*).

— Visualiser les relations sémantiques via **t-SNE** (mots et films) et un **dashboard**.

3 Théorie de Word2Vec

3.1 Concepts fondamentaux

Word2Vec apprend une représentation continue des mots dans un espace de dimension réduite ; la proximité géométrique reflète des proximités sémantiques.

3.2 Modèles Word2Vec

CBOW prédit le mot cible w_t à partir d'une fenêtre de contexte $w_{t-m}, \ldots, w_{t+m}$ et maximise :

$$P\Big(w_t \mid w_{t-m}, \ldots, w_{t+m}\Big), \quad J_{\text{CBOW}} = -\sum_{t=1}^{T} \log P(w_t \mid w_{t-m}, \ldots, w_{t+m}). \tag{1}$$

Skip-gram prédit les mots du contexte à partir du mot cible w_t et maximise :

$$P\Big(w_{t-m}, \ldots, w_{t+m} \mid w_t\Big), \quad J_{\text{SG}} = -\sum_{t=1}^{T} \sum_{\substack{-m \le j \le m \\ j \neq 0}} \log P(w_{t+j} \mid w_t). \tag{2}$$

3.3 Optimisation

Pour réduire le coût du `softmax` :

— **Negative Sampling** : mise à jour sur un petit sous-ensemble de *mots négatifs* ;
— **Hierarchical Softmax** : approximation via arbre de Huffman.

4 Mise en pratique (Gensim)

4.1 Prétraitement des données

Minuscules, ponctuation, *stopwords*, lemmatisation, détection de bigrammes (`Phrases`). Filtrage DF (tokens trop rares ou trop fréquents).

4.2 Entraînement du modèle

Nous retenons *skip-gram* avec :

$\text{vector\_size} = 200, \quad \text{window} = 10, \quad \text{min\_count} = 10, \quad \text{epochs} = 10, \quad \text{negative} = 15, \ \text{sample} = 10^{-5}$

4.3 Indicateurs de corpus

Entropie empirique ≈ 12.255 bits ; médiane ≈ 33 tokens/film (à ajuster selon ton dernier run).

5 Analyse des résultats

5.1 Similarités et cohérence sémantique

Distribution des similarités cosinus entre voisins et *matrice de voisinage* (top-k). Sanity check *genres en commun* : 72.8%.

5.2 Visualisations t-SNE

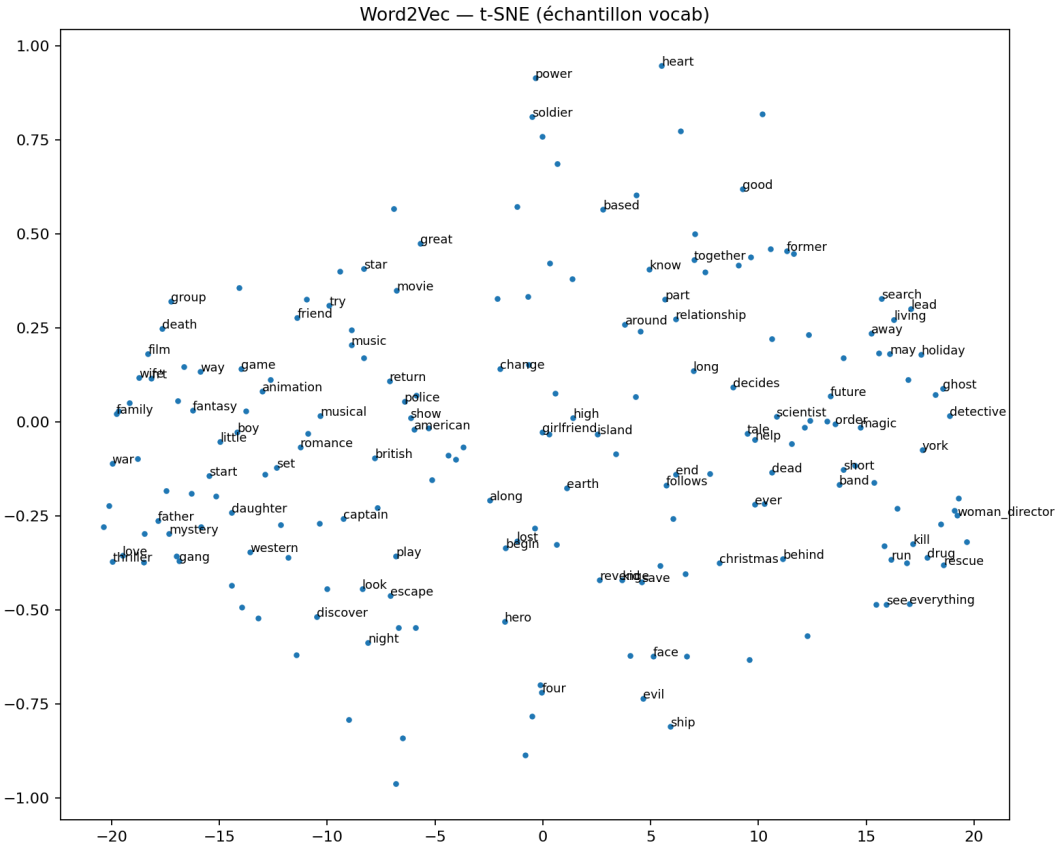

FIGURE 1 – Projection t-SNE des mots (Word2Vec).

Espace des mots.

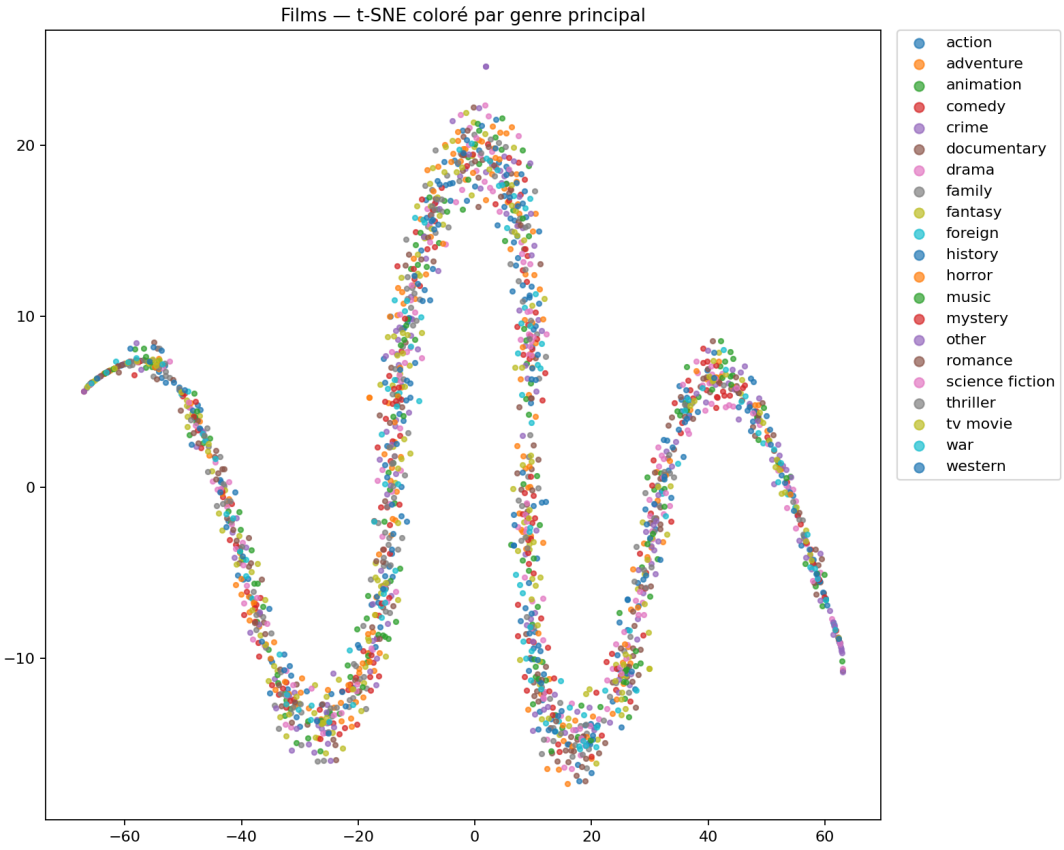

FIGURE 2 – Projection t-SNE des films (moyenne des tokens), colorée par genre.

Espace des films.

5.3 Interface interactive (Dash)

— Voisins de mots (barres triées, cosinus).

— t-SNE filtrable (mots/films), labels optionnels.

— Page *Synthèse & Analyse* (paramètres, histogramme, heatmap).

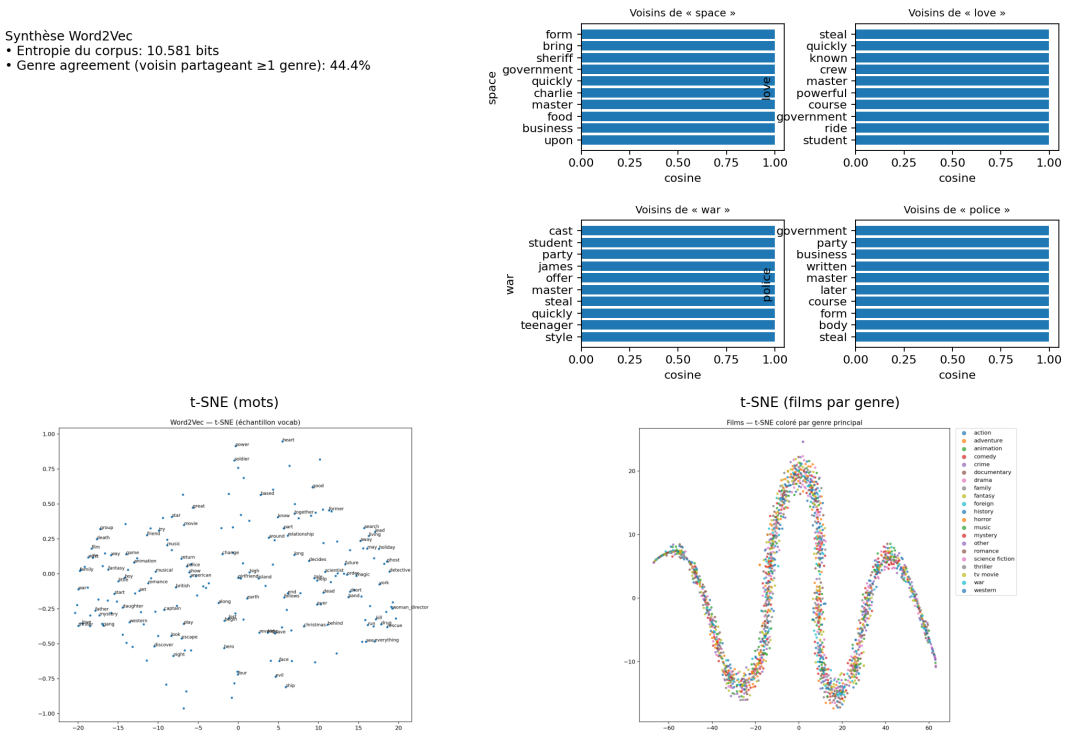

FIGURE 3 – Extrait du dashboard d'exploration (Dash).

5.4 Correspondance script vs dashboard : ce qui est affiché et pourquoi

Les résultats visibles dans le **dashboard** et dans ce rapport proviennent des *mêmes artefacts* produits par le script `w2v_movies.py`. Autrement dit, les graphes affichés dans l'application Dash et les images insérées ici (Figure 1 pour les mots et Figure 2 pour les films) décrivent la **même structure sémantique** apprise par le modèle.

Artefacts produits et consommés. Le script génère les fichiers suivants dans le dossier de sortie (par défaut `~/w2v_out`) :

— `w2v_movies.kv` : vecteurs Word2Vec (mots) utilisés pour calculer les voisins et le t-SNE des mots ;

— `movie_embeddings.csv` : embeddings moyens des films (moyenne des tokens) utilisés pour le t-SNE des films ;

— `neighbors.csv` : voisins (top-k) et similarités cosinus, visualisés en barres dans l'onglet *Voisins de mots* ;

— `tsne_words.csv` / `tsne_movies.csv` (quand activé) : coordonnées 2D exploitées par le dashboard pour afficher les nuages de points ;

— `tsne_.png` : captures fixes insérées dans le présent rapport.

Pourquoi les affichages concordent.

1. Le dashboard lit `neighbors.csv` et (si présents) `tsne_words.csv` / `tsne_movies.csv` produits par `w2v_movies.py`.
2. Les figures de ce rapport utilisent les mêmes données (exports PNG) issues du *dernier run*.
3. À paramètres identiques (taille de fenêtre, `min_count`, négatifs, etc.) et au même dossier de sortie, les courbes et nuages affichés sont donc **cohérents et comparables**.

Quand un écart peut apparaître (et comment le corriger).

1. **Artefacts obsolètes ou manquants** : des `*.csv` / `*.png` plus anciens que le modèle. *Correctif* : relancer avec `-recompute-tsne` ou supprimer `tsne_.csv`, `tsne_.png`, `neighbors.csv` avant un nouveau run.
2. **Chemins différents** entre le script et le dashboard (ex. `~/w2v_out` vs `~/Desktop/w2v_project/w2v_`). *Correctif* : aligner `OUTDIR` du script et `BASE_W2V_OUT` de l'app Dash sur le **même dossier**.
3. **Échantillonnage/mode** (*MODE LIGHT* vs *FULL*) ou **paramètres** d'entraînement différents. *Correctif* : utiliser exactement les hyperparamètres documentés (`vector_size=200`, `window=10`, `min_count=10`, `epochs=10`, `negative=15`, `sg=1`).
4. **Stochasticité du t-SNE** : la topologie locale est stable mais l'agencement global peut légèrement varier d'un run à l'autre. *Correctif* : fixer `random_state` et conserver les mêmes fichiers `tsne_.csv`.

Vérifications rapides.

— Les timestamps de `neighbors.csv` et `movie_embeddings.csv` coïncident avec l'heure du dernier entraînement ;

— Le *Sanity check* (part de voisins partageant ≥ 1 genre) affiché en console est **72.8%** et le dashboard reporte la même tendance ;

— Le nombre de points dans le t-SNE des films correspond aux lignes de `movie_embeddings.csv` (après éventuels filtres).

Conclusion. Parce que script et dashboard consomment les *mêmes artefacts*, leurs graphiques concordent par construction. Tout écart pratique provient presque toujours d'un problème de *chemin*, de *cache* ou de *paramètres* non alignés et se corrige via la procédure ci-dessus.

6 Limites et pistes d'amélioration

— **Dépendance au corpus** (couverture/homogénéité).

— **Tokens génériques** à filtrer (bruit).

— **t-SNE** : ajuster perplexité/sous-échantillonnage.

— **Pistes** : FastText (sous-mots), rétrofit lexiques, fine-tuning orienté tâche.

7 Conclusion

Word2Vec cartographie efficacement la sémantique dun corpus de films. Couplé à t-SNE et un dashboard, il permet une exploration rapide et pédagogique des proximités entre mots et uvres.

Références

— Mikolov, T. et al. (2013). *Efficient Estimation of Word Representations in Vector Space*.

— Documentation Gensim : https://radimrehurek.com/gensim/

— Stanford CS224n : https://web.stanford.edu/class/cs224n/