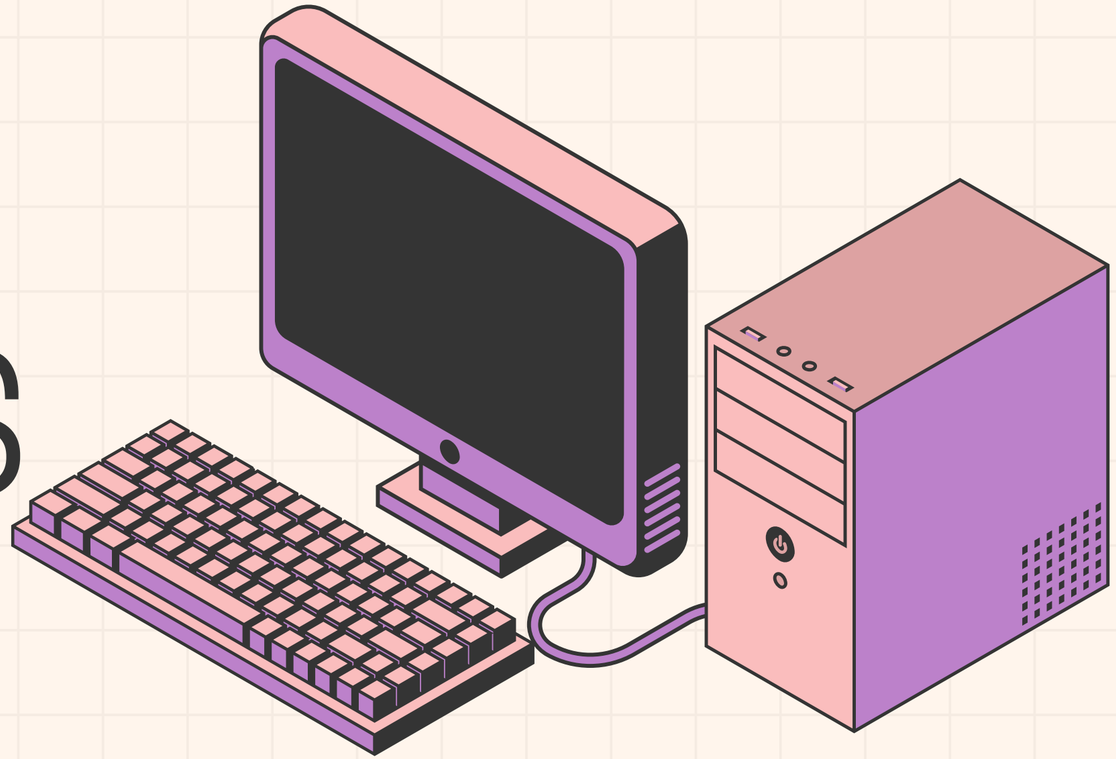
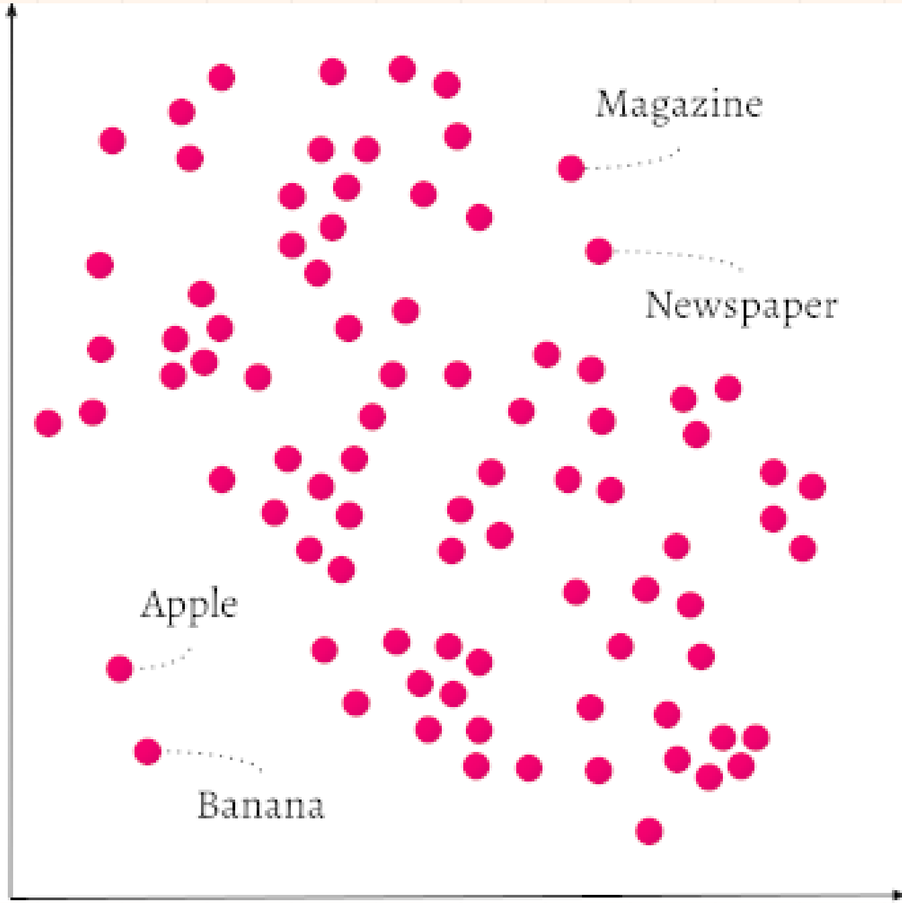
LES BDD
VECTORIELLES



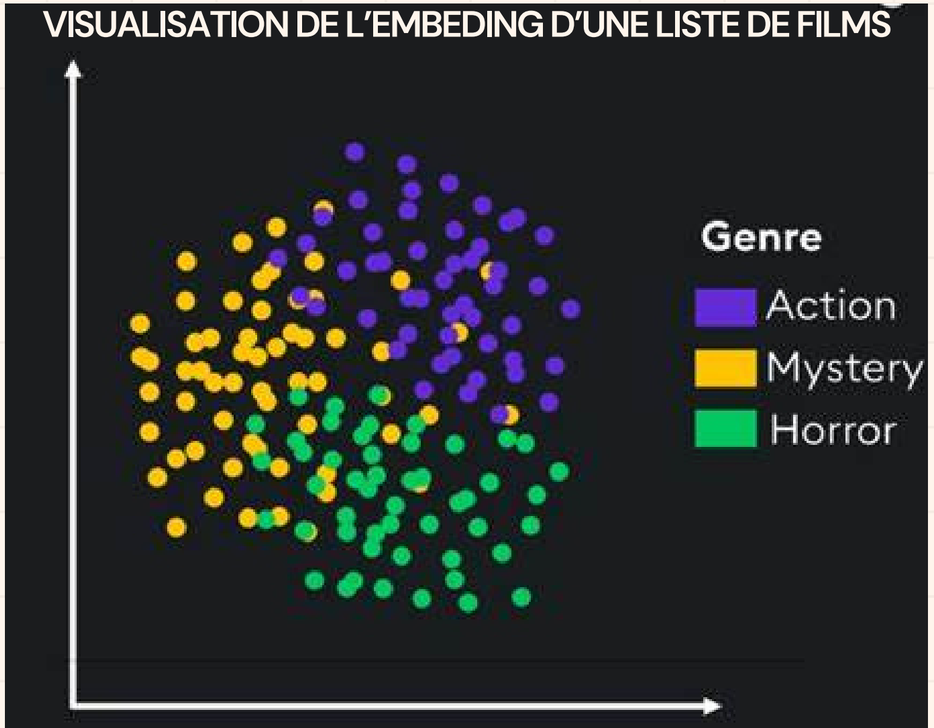
DÉFINITION

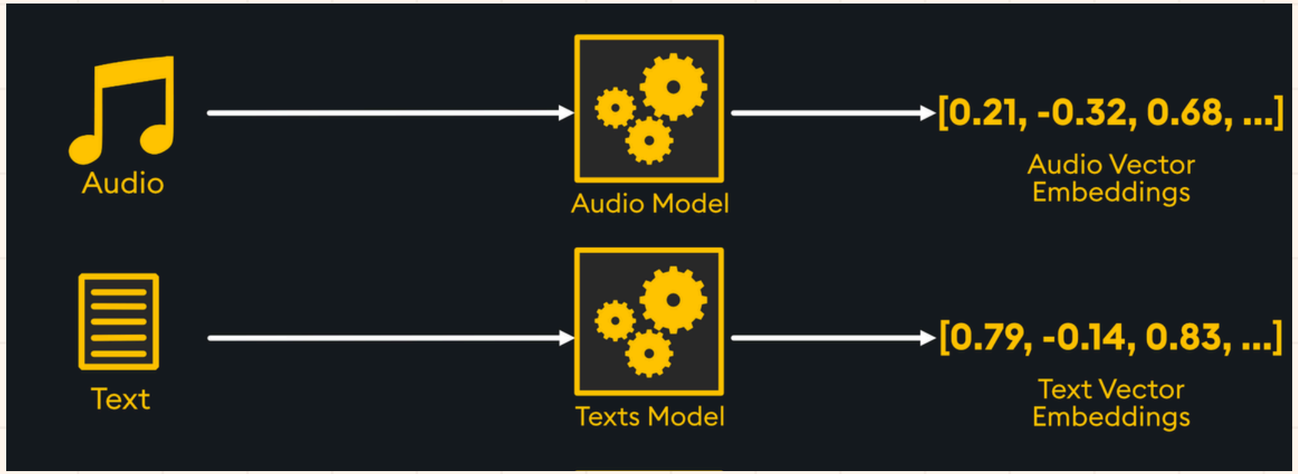
- Les bases de données (BDD) vectorielles stockent les éléments d'information sous forme de vecteurs.
- Elles sont souvent utilisées pour optimiser les modèles d’apprentissage automatique.
- Elles regroupent les éléments apparentés, permettant des requêtes plus rapides et rentables et évitent de soumettre constamment des données au modèle.
- Elles sont idéales pour des applications comme la recherche sémantique et la détection d'anomalies.



VECTORISATION (EMBEDDING)

Un embedding vectoriel désigne un vecteur créé comme représentation numérique d'objets de données généralement non numériques.

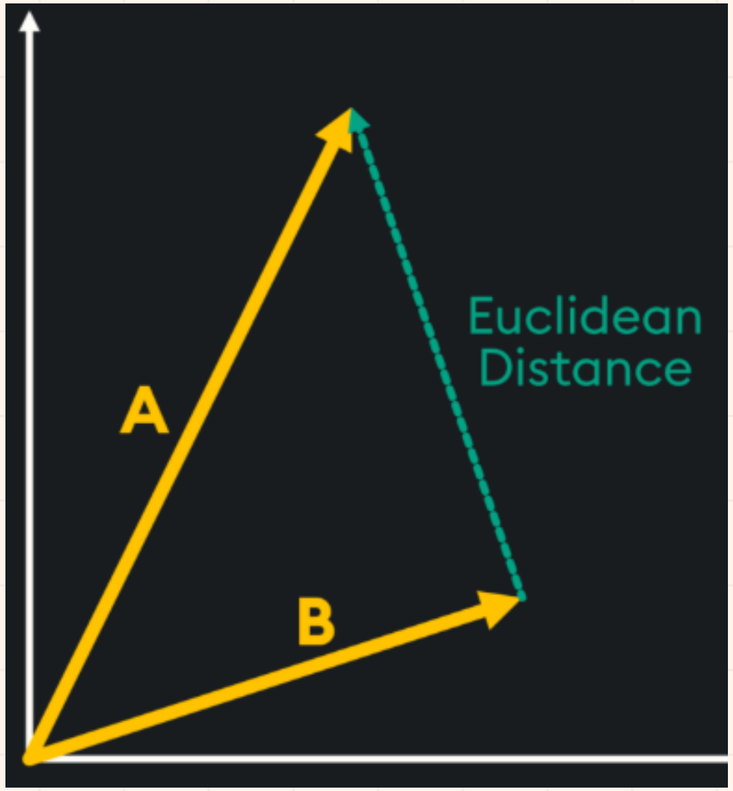




En général, les embeddings visent à capturer des informations sémantiques, contextuelles ou structurelles pertinentes pour la tâche spécifique à accomplir.

COMPARAISON VECTORIELLE

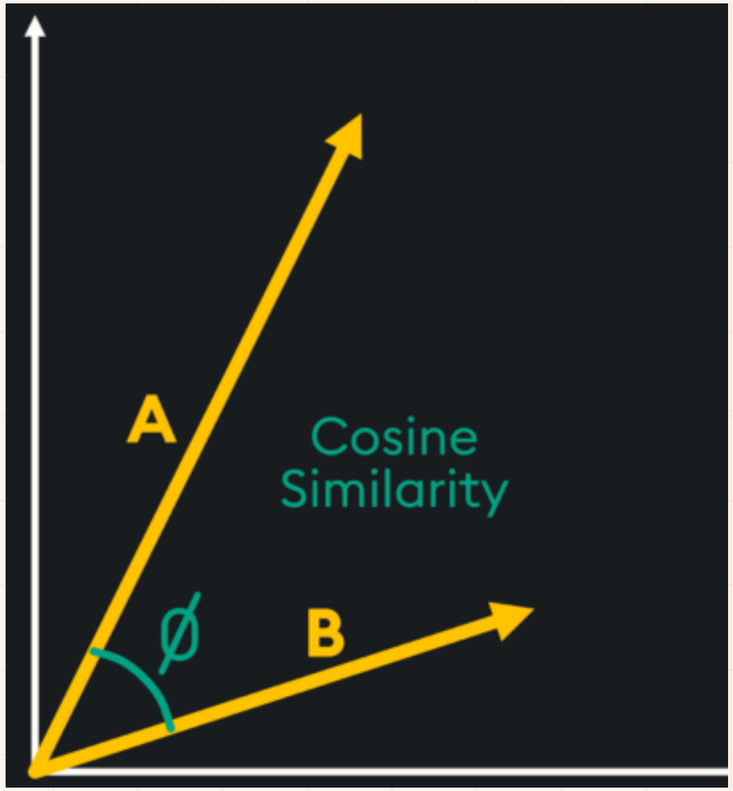
Distance Euclidienne



Plus la distance est oroche de 0, plus les vecteurs sont similaires sémantiquement

Ex : Analyse de clusters

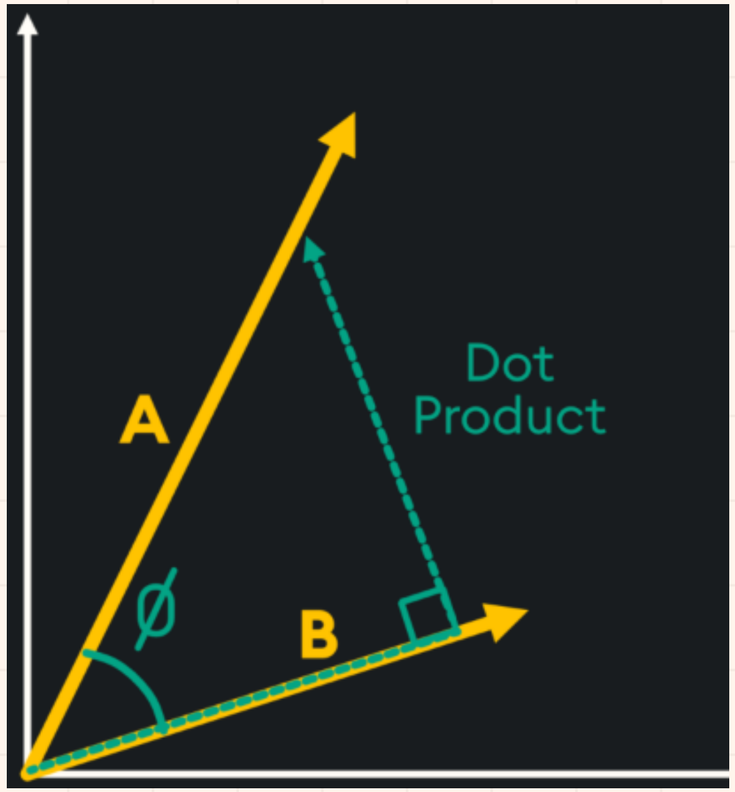
Similarité cosinus



Les vecteurs sont moins similaires plus l'angle se rapproche de 180 et plus similaires si l'angle se rapproche de 0

Ex : Comparaison de documents

Produit scalaire

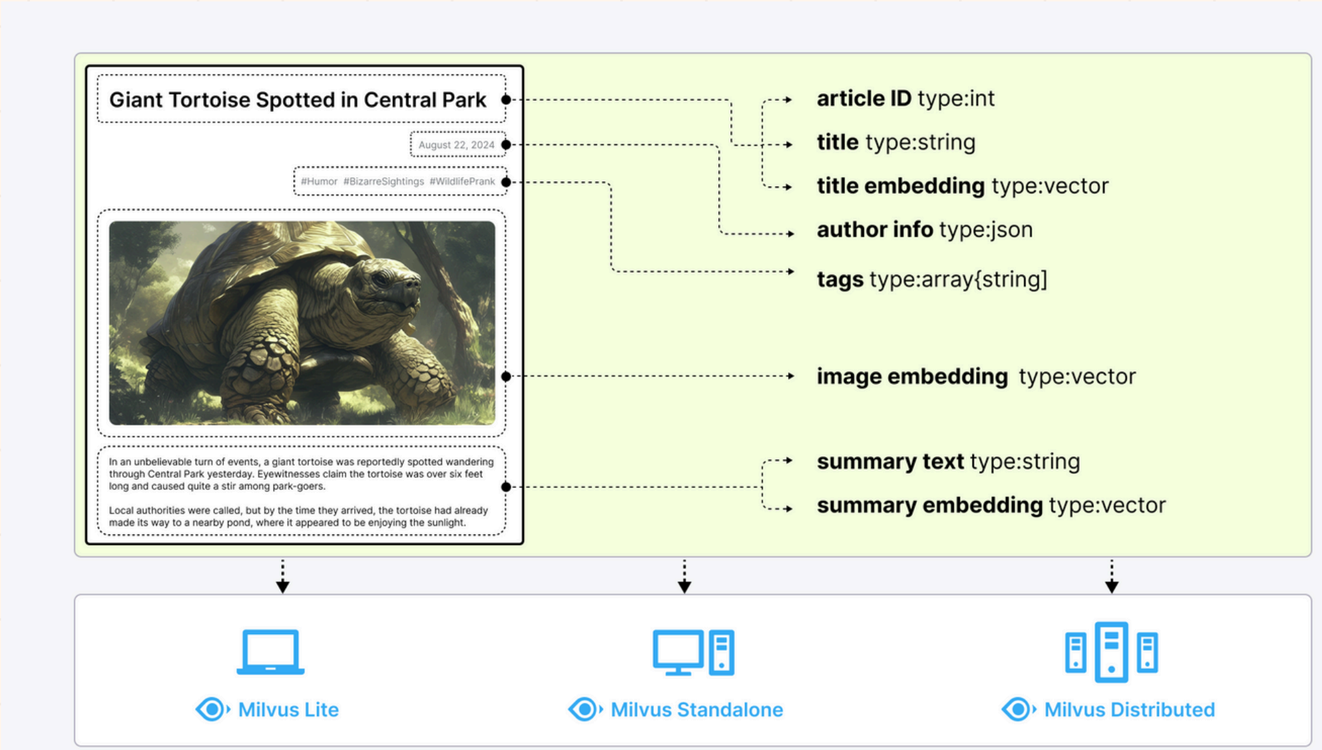


angle 0 : similaires
angle 180 : opposés
angle 90 : sans rapport

Ex : Récupération et correspondance d'images

Milvus

- BDD vectorielle créée en 2019 par Zilliz
- Projet **opensource** sous la license Apache 2.0
- 3 modes de déploiement:
  - **Lite** : Librairie Python (utilisée dans notre tuto)
  - **Standalone** : Déploiement sur une seule machine, avec image Docker regroupant tous les composants
  - **Distributed** : Adapté au déploiement sur clusters Kubernetes, adopte une architecture cloud-native. Garantit une redondance des composants critiques.

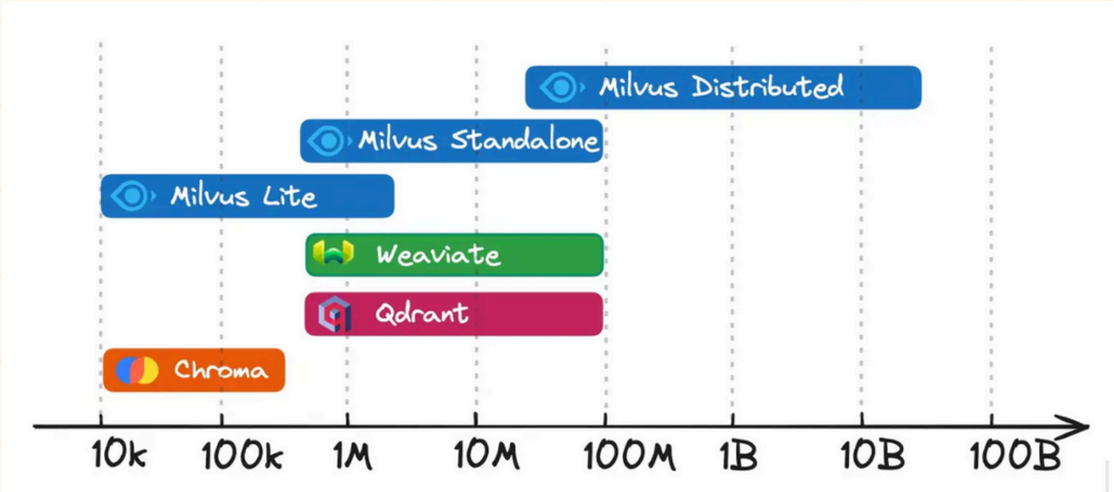


AVANTAGES DE MILVUS

- 2 à 5 fois plus rapide que d’autres BDD Vectorielles
- **Scalable** horizontalement grâce à son architecture cloud-native et découplée

les trois tâches les plus critiques - la recherche, l'insertion de données et l'indexation/compactage - sont conçues comme des processus facilement parallélisables.

- Gère des types de données avancés (en plus des types primitifs) : JSON, Distance Matrix…
- Supporte de **nombreux types de recherche**; et des hybrid search. Permet aussi la configuration de l’indexation
- Possède de nombreux SDKs: Python, Java, Go, C++, Node.js, et Ruby



TUTORIELS