Vector Search

06: Vector search - IPSA - Hiver 2025
7/03/2025 - 4h

Last time on NoSQL

- MongoDB - schema patterns
- projets

Cette semaine

Vector search

- NLP
- embeddings des textes: principes et méthodes
- score de similarité
- vectoriser du texte, APIs et modèles
- Bdds vectorielles
- RAG
- Vector search et MongoDB

Préhistoire

tf-idf : matrice word - document.

- vocabulaire fixé à l'avance
- preprocessing du texte: normalisation, lemmatisation, stopwords, …
- occurrence des mots / taille du vocabulaire

Très efficace pour la classification simple: détecter le spam, analyse de sentiment etc. Toujours utilisé pour sa rapidité.

Token

Quel unité pour le texte : mots, syllabes, lettres ?

l'unité est appelé **token**. On parle de tokenisation

- lettre : correcteur orthographique type smartphone
- mots : vocabulaire très grand et problèmes de out of vocabulary
- ~syllabe : flexible, vocabulaire plus petit

Comment passer du texte au vecteur ?

Entraîne un réseau de neurones à prédire le mot manquant dans un corpus de phrases - principe des LLMs et des transformers

On enregistre les coefficients de la dernière couche du réseau de neurones

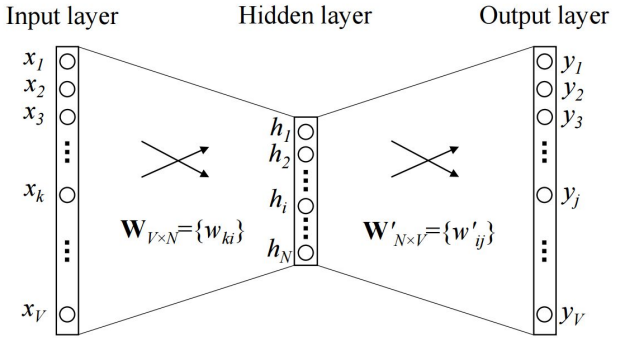
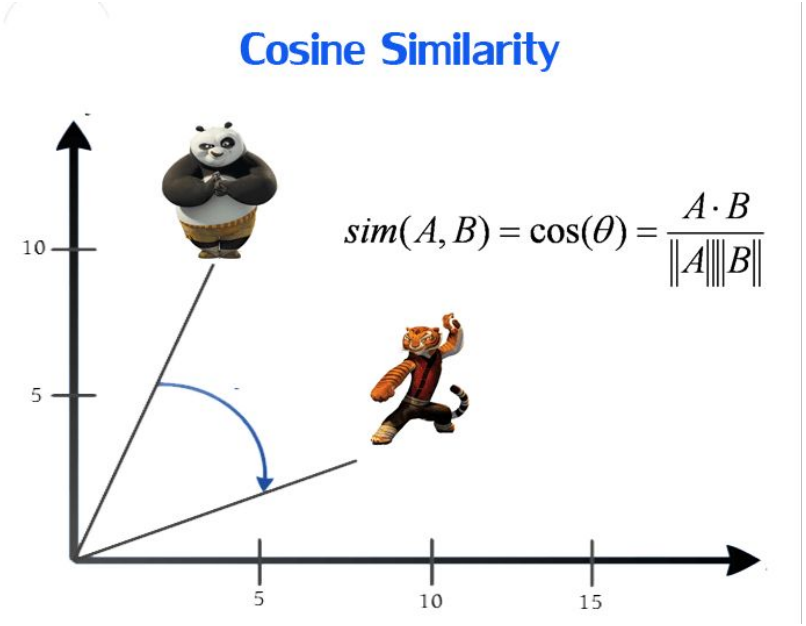


Figure 1: A simple CBOW model with only one word in the context

2015 : CBOW, Word2Vec, Glove, ...

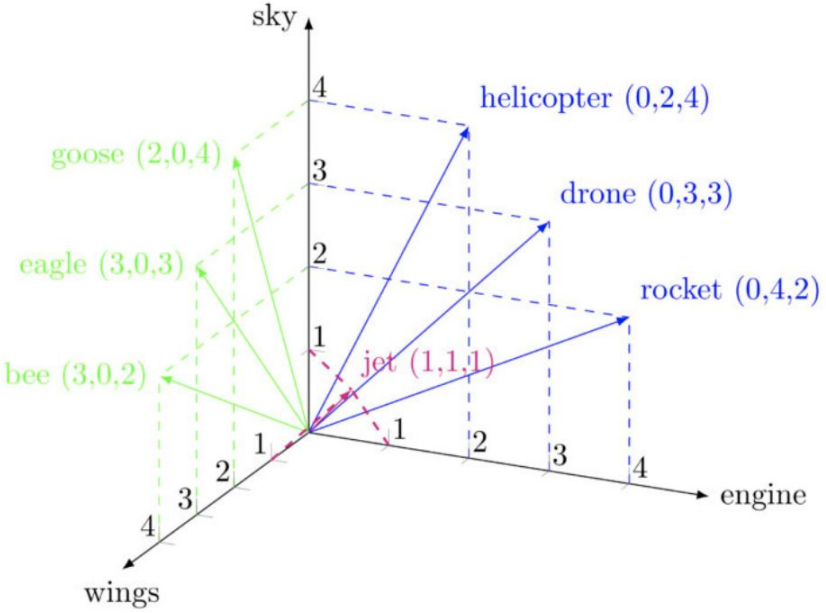
Distance entre 2 textes

Transformer un texte en un équivalent vecteur permet d'avoir une distance entre 2 textes



Embeddings

Tout est vecteur



mots => vecteurs

- poulet -> [1,4,5,-2, 7, … ,8]
- banane -> [2,-3, 5.3, 12, 127, …. -0.1]
- chat -> [1,2,3,, ….,]

words, sentences, paragraphs, books, …

semantic relationship

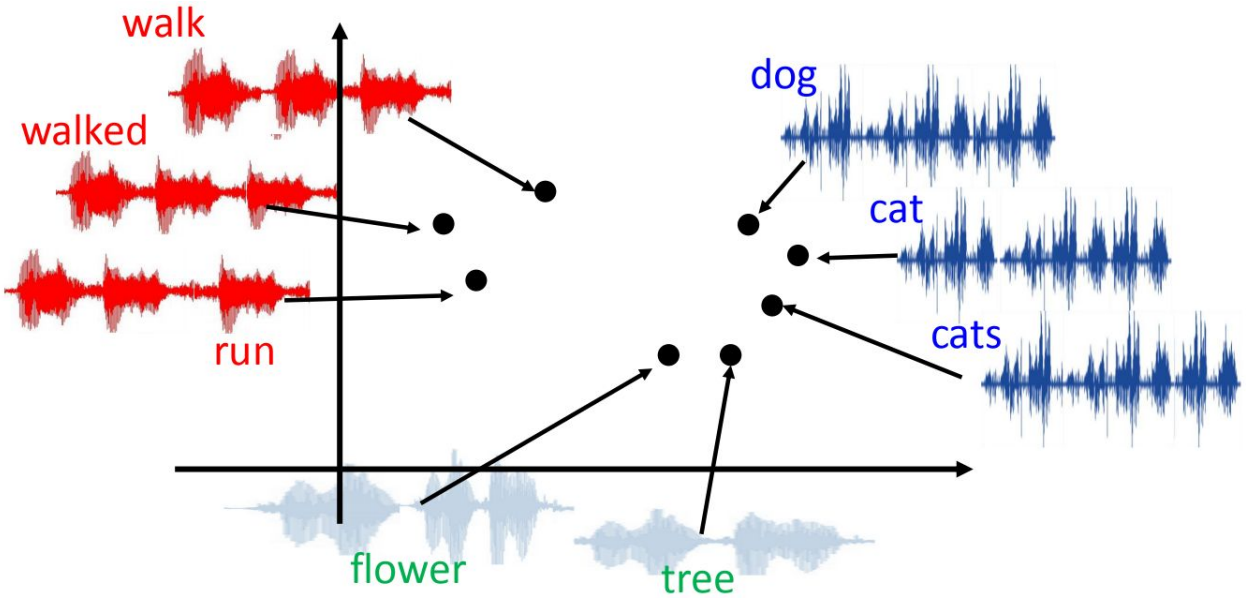
- king - man = queen - woman

distance between text

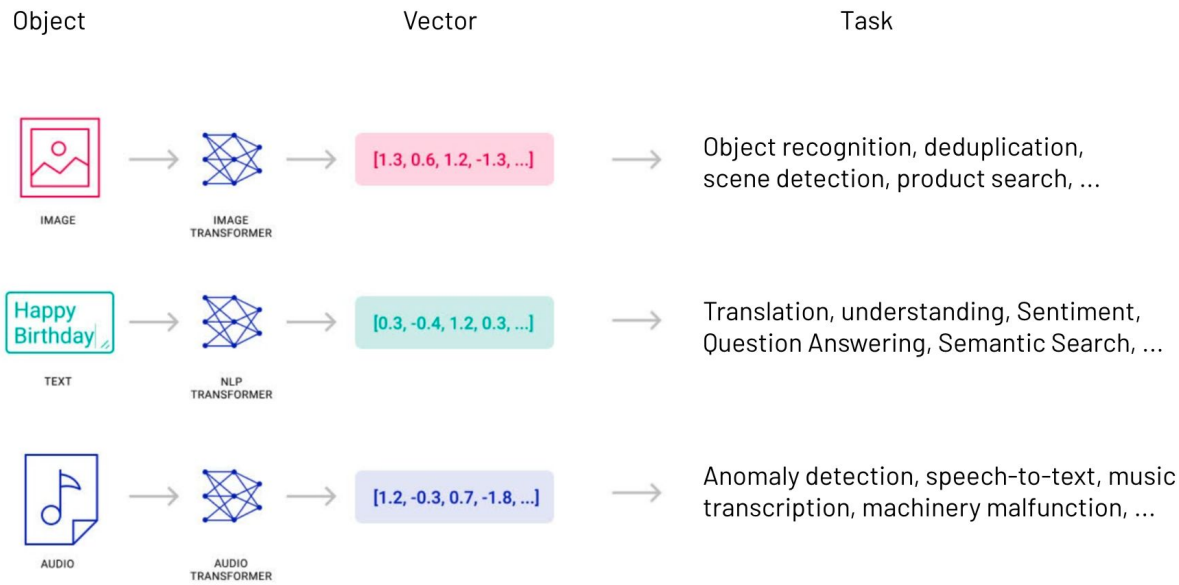
- d(cat, dog) < d(cat, banana)

We are all vectors now

Audio Embeddings



Multimodal



Must read: [The Multimodal Evolution of Vector Embeddings]

Comment vectoriser un texte ?

Trouver un modèle sur huggingface :

MTEB Leaderboard - a Hugging Face Space by mteb

- langue(s)
- multi modal ou texte

Rank (Borda)	Model	Zero-shot	Number of Parameters	Embedding Dimensions	Max Tokens	Mean (Task)	Mean (TaskType)	Bitext Mining	Classification
11	SFR-Embedding-2_R	✅	7B	4096	32768	59.46	52.62	68.84	59.01
1	gte-Qwen2-7B-instruct	⚠️	7B	3584	131072	62.09	55.68	73.92	61.55
10	gte-Qwen1.5-7B-instruct	⚠️	7B	4096	32768	58.58	52.28	60.8	58.24
5	SFR-Embedding-Mistral	✅	7B	4096	32768	60.46	53.68	70	60.02
9	gte-Qwen2-1.5B-instruct	⚠️	1B	8960	131072	58.81	52.24	62.51	58.32
2	Linq-Embed-Mistral	⚠️	7B	4096	32768	61.04	53.9	70.34	62.24
7	e5-mistral-7b-instruct	✅	7B	4096	32768	59.79	52.81	70.58	60.31

Exemple Mistral 7B - smaller MiniLM-L6-v2

https://huggingface.co/intfloat/e5-mistral-7b-instruct (>6Gb)

https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

Colab :

- Sentence Embeddings https://colab.research.google.com/drive/1PJfgennLBGUUdAGOLR3hy9FYx5Aub0XD#scrollTo=l5b_1iimTUKF

Use an API

OpenAI API : https://platform.openai.com/docs/guides/embeddings

Mistral API: https://docs.mistral.ai/capabilities/embeddings/

```
import os
from mistralai import Mistral

api_key = os.environ["MISTRAL_API_KEY"]
model = "mistral-embed"

client = Mistral(api_key=api_key)

embeddings_batch_response = client.embeddings.create(
    model=model,
    inputs=["Embed this sentence.", "As well as this one."],
)
```

Coûts vectorisation via API

OpenAI :
https://platform.openai.com/docs/guides/embeddings#embedding-models

0.0016 cts / page

Usage is priced per input token. Below is an example of pricing pages of text per US dollar (assuming ~800 tokens per page):

MODEL	~ PAGES PER DOLLAR	PERFORMANCE ON MTEB EVAL	MAX INPUT
text-embedding-3-small	62,500	62.3%	8191
text-embedding-3-large	9,615	64.6%	8191
text-embedding-ada-002	12,500	61.0%	8191

Applications des embeddings textuels

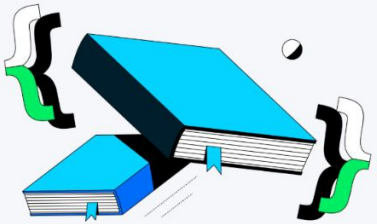
- Classification : Catégorisation de textes
- clustering : identification de groupes naturels
- Systèmes de recommandation: Suggestion de contenu similaire
- Recherche d'information: Amélioration de la pertinence des recherches
- Détection d'anomalies: Identification de documents atypiques
- Construction de graphes de connaissances: Création de relations structurées
- Transfer learning: Utilisation comme caractéristiques pour d'autres tâches d'IA
- Applications multi-modales: Connexion du texte avec images, audio ou vidéo
- Reconnaissance d'intention: Compréhension des requêtes utilisateur
- Compression sémantique: Représentation compacte de documents
- Apprentissage zéro-shot/few-shot: Fonctionnement avec peu d'exemples

Enbeddings dans MongoDB

Unlock the power of AI with MongoDB

Discover how to leverage MongoDB to streamline development for the next generation of AI-powered applications.

Visit the AI Learning Hub >



Introduction to AI and Vector Search

Sign up for our free MongoDB University course to learn about the foundations of AI.

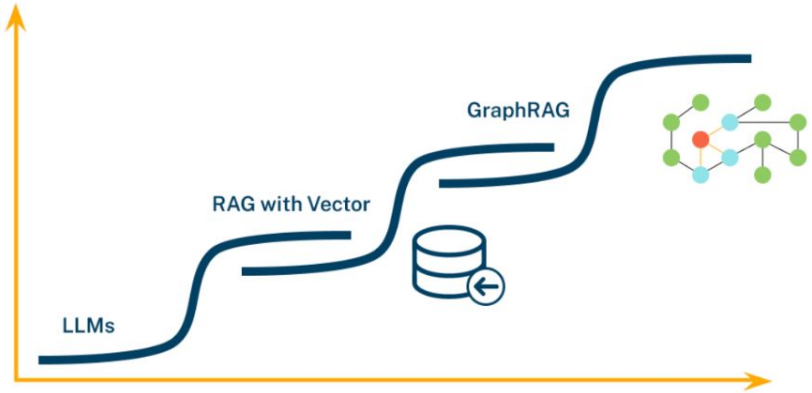
Start learning today >

How to Choose the Best Embedding Model for Your LLM Application

Read our tutorial to learn the fundamentals of building a RAG application, starting with choosing the right embedding model.

Read the tutorial >

Enbeddings dans Neo4j



Accurate, Explainable GenAI With GraphRAG

Provide GenAI responses grounded in your real-world data and relationships. GraphRAG combines knowledge graphs with Retrieval-Augmented Generation (RAG), enabling you to build GenAI applications that deliver more accurate, relevant, and explainable results.

Learn How ›

Vector store

What's a vector store

Un **vector store** est une base de données spécialisée conçue pour stocker, indexer et rechercher efficacement des vecteurs de haute dimension.

Europe:

- Weaviate.io – Open-source vector search engine with built-in ML models. (Netherlands)
- vespa.ai (Norway)

Et USA

- Pinecone – Scalable, fully managed vector database.
- FAISS (by Facebook) – Fast indexing for large-scale search.
- Chroma – Lightweight vector store often used in RAG pipelines.

Utiliser un Vector store dédié vs bdd SQL ou NoSQL + vector search

Les Bdd vectorielles

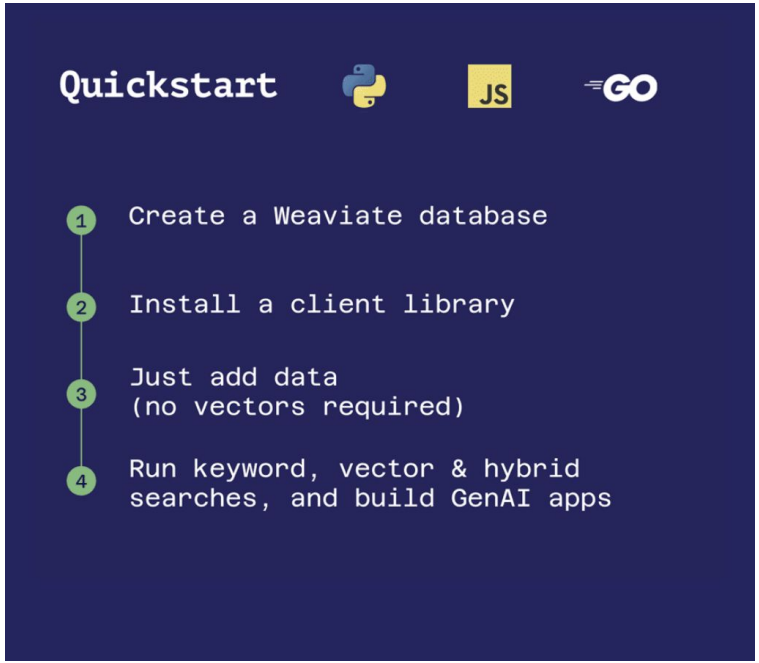
- **Performances et scalabilité** : optimisées pour la recherche de similarité à grande échelle.
- **Fonctionnalités spécialisées** : supportent nativement les opérations vectorielles (addition, soustraction, …) et les recherches hybrides (vecteur + scalaire).
- **Intégration simplifiée** : APIs et SDK intuitifs facilitent l'intégration avec les workflows de Machine Learning.

Vector stores

Feature	Pinecone	Weaviate	Milvus	Vespa	Qdrant
Type	Managed	Self/Cloud	Self-hosted	Self-hosted	Self-hosted
Open-source	No	Yes	Yes	Yes	Yes
Scale (billions)	Medium	Medium	High	High	Medium
Hybrid Search	Yes	Yes	Yes	Yes	Yes
Ease of Use	High	Medium	Medium	Low	High
Integration (AI)	Strong	Strong	Medium	Medium	Medium
Cost (self-host)	High	Low	Low	Medium	Low

weaviate

- weaviate.io
- create a cluster
- connect
-



weaviate - add vectors

Install

- https://weaviate.io/developers/weaviate/installation
- ou https://weaviate.io/developers/weaviate/quickstart

- https://weaviate.io/developers/weaviate/quickstart/local
- get a corpus - dataset
- embed the text et Bring your own vectors
- ou import data with vectors

What can you do ?

we have text and vectors in a weaviate database

what's next ?

- `with_near_text` and `with_hybrid` search
- anomaly detection for instance written and spoken style
- Multi-Modal Search

Vector search in MongoDB

Vector search dans MongoDB

reprendre votre cluster dans Atlas
et le sample dataset avec la collection embedded movies

Vector search dans MongoDB

```
db.embedded_movies.aggregate([
  {
    "$vectorSearch": {
      "index": "vector_index",
      "path": "plot_embedding",
      "queryVector": [-0.0016261312,-0.028070757,..., .009710136],
      "numCandidates": 150,
      "limit": 10,
      "quantization": "scalar"
    }
  },
  {
    "$project": {
      "_id": 0,
      "plot": 1,
      "title": 1,
      "score": { $meta: "vectorSearchScore" }
    }
  }
])
```

Pratique

Projet : base interrogation d'un corpus - MongoDB

- load un dataset avec du texte dans une base MongoDB
- Vectorizer le texte (python, node.js)
- ajouter les vecteurs a la base

Puis faire une interface web d'interrogation de la base

- input query
- vectorizes the input query
- finds the N closest match from collection / table
- displays N closest match

Interrogation d'un corpus - Weaviate

load un dataset avec du texte dans une base Weaviate

- install weaviate en local ou creer un cluster
- setup weaviate pour utiliser le module **text2vec-huggingface**
- ou declarer votre cle API
- la vectorisation est faite en meme temps que l'ingestion des textes

Puis faire une interface web d'interrogation de la base

- input query
- vectorizes the input query
- finds the N closest match from collection / table
- displays N closest match

RAG

- get wikipedia page on topic : LoL, art, …
- embed each section
- add to collection
- query collection
- prompt + add retrieved text
- get answer
- https://docs.google.com/document/d/1k1JQ7nCgM2cO5P_WUPgkXwaQDJYSk365Gg1kI6zncgQ/edit?tab=t.0