**🔍 À quoi sert sentence-transformers ?**

Elle permet de transformer une **phrase, un paragraphe ou un document** en un **vecteur sémantique**, c’est-à-dire une représentation mathématique qui capture le **sens** du texte.

**📦 Basée sur quoi ?**

Elle est basée sur :

* les modèles **BERT**, **RoBERTa**, **DistilBERT**, etc.
* avec un fine-tuning spécial pour que **les phrases proches aient des vecteurs proches** dans l’espace vectoriel.

**✅ Utilisations principales :**

| **Cas d’usage** | **Exemple** |
| --- | --- |
| Recherche sémantique | Trouver un document qui a du **sens proche** d’une requête |
| RAG / Chatbots IA | Créer des vecteurs à partir de documents pour un chatbot intelligent |
| Clustering | Regrouper automatiquement des textes similaires |
| Détection de duplicats | Identifier des phrases qui disent la même chose |
| Recommandation | Suggérer des textes/articles similaires |

**📌 Exemple rapide en code**

from sentence\_transformers import SentenceTransformer

model = SentenceTransformer('all-MiniLM-L6-v2') # modèle rapide et léger

sentence = "L'IA médicale révolutionne les diagnostics."

embedding = model.encode(sentence)

print(embedding.shape) # (384,) — vecteur dense de 384 dimensions

**💡 Comparé à OpenAI Embeddings ?**

| **Critère** | **sentence-transformers** | **openai.embeddings** |
| --- | --- | --- |
| Local / Cloud | 📥 Local (gratuit) | ☁️ Cloud (payant via API) |
| Vitesse | ⚡ Rapide en local | Plus lent via API |
| Coût | Gratuit | Payant |
| Qualité | Bonne | Excellente (selon modèle) |

### 🧠 📦 **Librairies principales**

| **Package** | **Rôle** |
| --- | --- |
| **langchain** | 🧠 Framework principal pour connecter LLMs + outils (vector DB, docs, etc.) |
| **langchain-pinecone** | 🔌 Intégration directe avec Pinecone (DB vectorielle) |
| **langchain-community** | 🔌 Intègre des connecteurs open-source de la communauté (FAISS, HuggingFace, etc.) |
| **langchain-openai** | 🔌 Connexion à OpenAI pour embeddings + chat complet |
| **langchain-experimental** | 🧪 Contient des modules "bêta" pour LLM agents, RAG avancé, etc. |

### 🗃️ **Pour la base de données vectorielle**

| **Package** | **Rôle** |
| --- | --- |
| **pinecone[grpc]** | 🚀 Client officiel Pinecone avec support gRPC (performances ++), pour stocker/rechercher des vecteurs |

### ⚙️ **Utilitaires et Flask**

| **Package** | **Rôle** |
| --- | --- |
| **flask** | 🌐 Framework web minimal en Python — tu crées une API ou une interface web |
| **python-dotenv** | 🔐 Permet de charger automatiquement tes variables d’environnement depuis un fichier .env |
| **pypdf** | 📄 Permet d’extraire le texte depuis des fichiers PDF (pour alimenter ton chatbot en contenu) |

## 🎯 Que peut-on construire avec tout ça ?

Avec ces librairies, tu peux construire un **chatbot capable de lire des documents PDF, de les vectoriser, de les stocker dans Pinecone, et de répondre aux questions avec GPT**. C’est une architecture RAG complète 💬📚.