La **recherche** à l'ère de l'**IA**

David Pilato | @dadoonet

Agenda

- La recherche "classique" et ses limites
- Modèle de ML et usages
- La recherche vectorielle ou hybride dans Elasticsearch
- OpenAI ChatGPT ou LLM avec Elasticsearch

Elasticsearch

You Know, for Search

Elasticsearch

Bordeaux JUG

elastic

These are not the droids
you are looking for.

```
GET /_analyze
{
  "char_filter": [ "html_strip" ],
  "tokenizer": "standard",
  "filter": [ "lowercase", "stop", "snowball" ],
  "text": "These are <em>not</em> the droids
           you are looking for."
}
```

```
These are <em>not</em> the droids you are looking for.
{ "tokens": [{
      "token": "droid",
      "start_offset": 27, "end_offset": 33,
      "type": "<ALPHANUM>", "position": 4
    },{
      "token": "you",
      "start_offset": 34, "end_offset": 37,
      "type": "<ALPHANUM>", "position": 5
    }, {
      "token": "look",
      "start_offset": 42, "end_offset": 49,
      "type": "<ALPHANUM>", "position": 7
    }]}
```

Recherche
sémantique
≠
Correspondance
littérale

Bordeaux JUG

AUJOURD'HUI

Escadron de vaisseaux X-wing

DEMAIN

Quel vaisseaux et équipages faut-il pour détruire une étoile de la mort quasi achevée ? Ou existe-t'il une faiblesse cachée ?

Bordeaux JUG

elastic

Elasticsearch

You Know, for **Vector** Search

Bordeaux JUG

elastic

Qu’est-ce qu’un **Vecteur** ?

Bordeaux JUG

elastic

Les embeddings représentent vos données

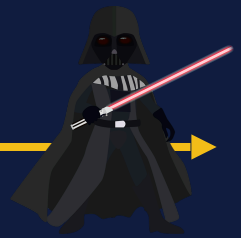
Exemple : vecteur 1 dimension

Réaliste

Dessin

Personnage	Vecteur
	[-1]
	[1]

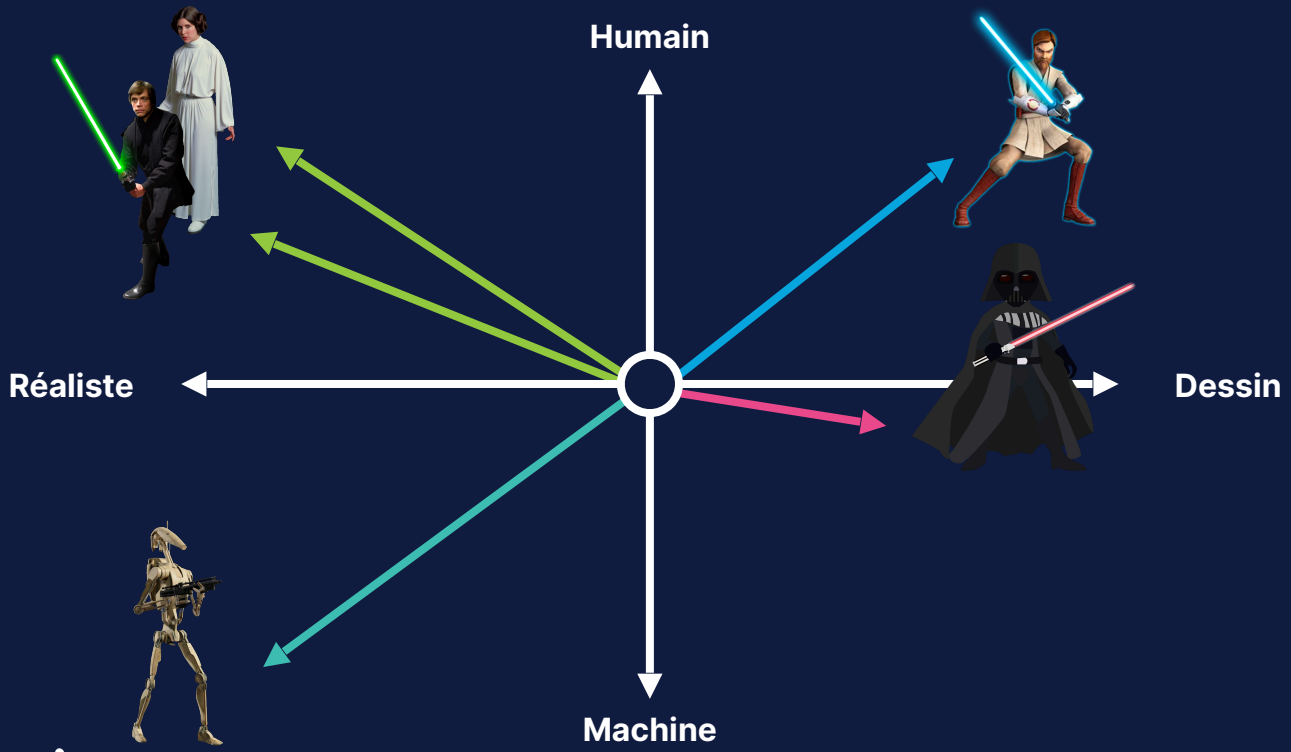
Plusieurs dimensions
pour représenter plusieurs aspects



Personnage	Vecteur
	[-1, 1]
	[1, 0]

Dans l’espace des embeddings

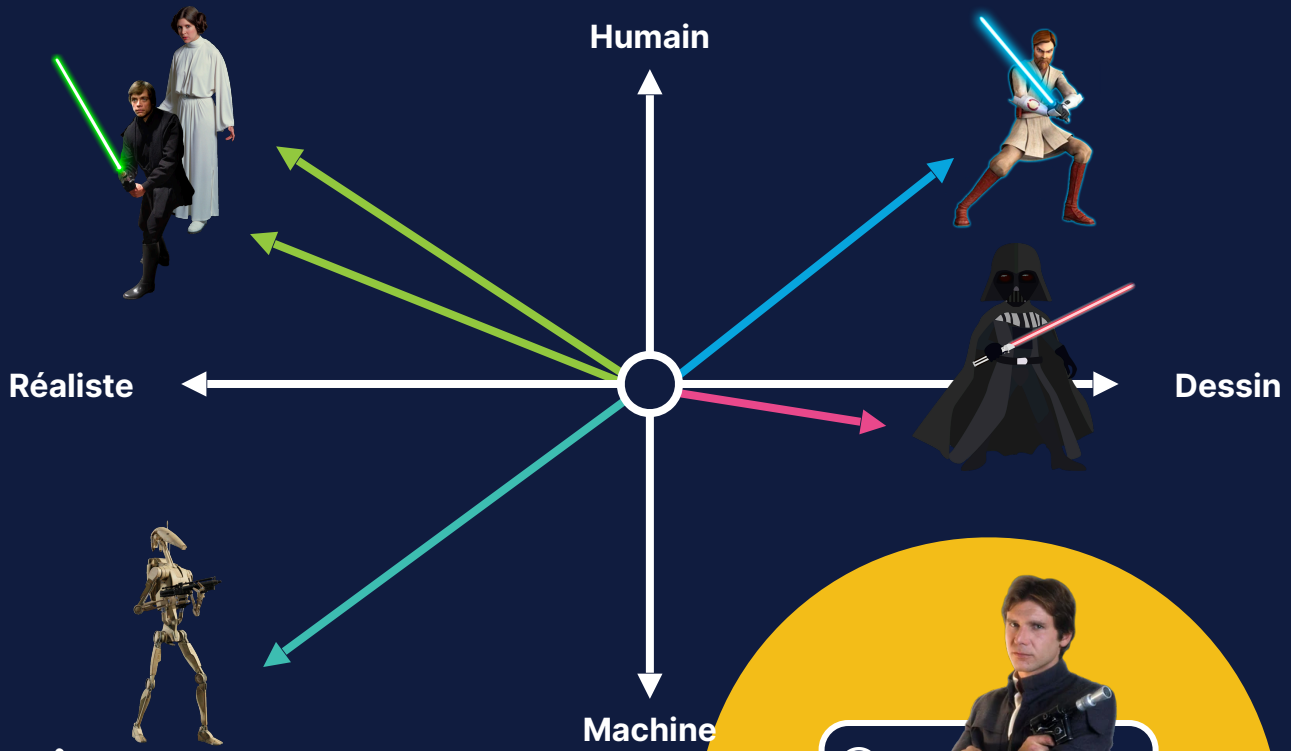
les données similaires sont regroupées



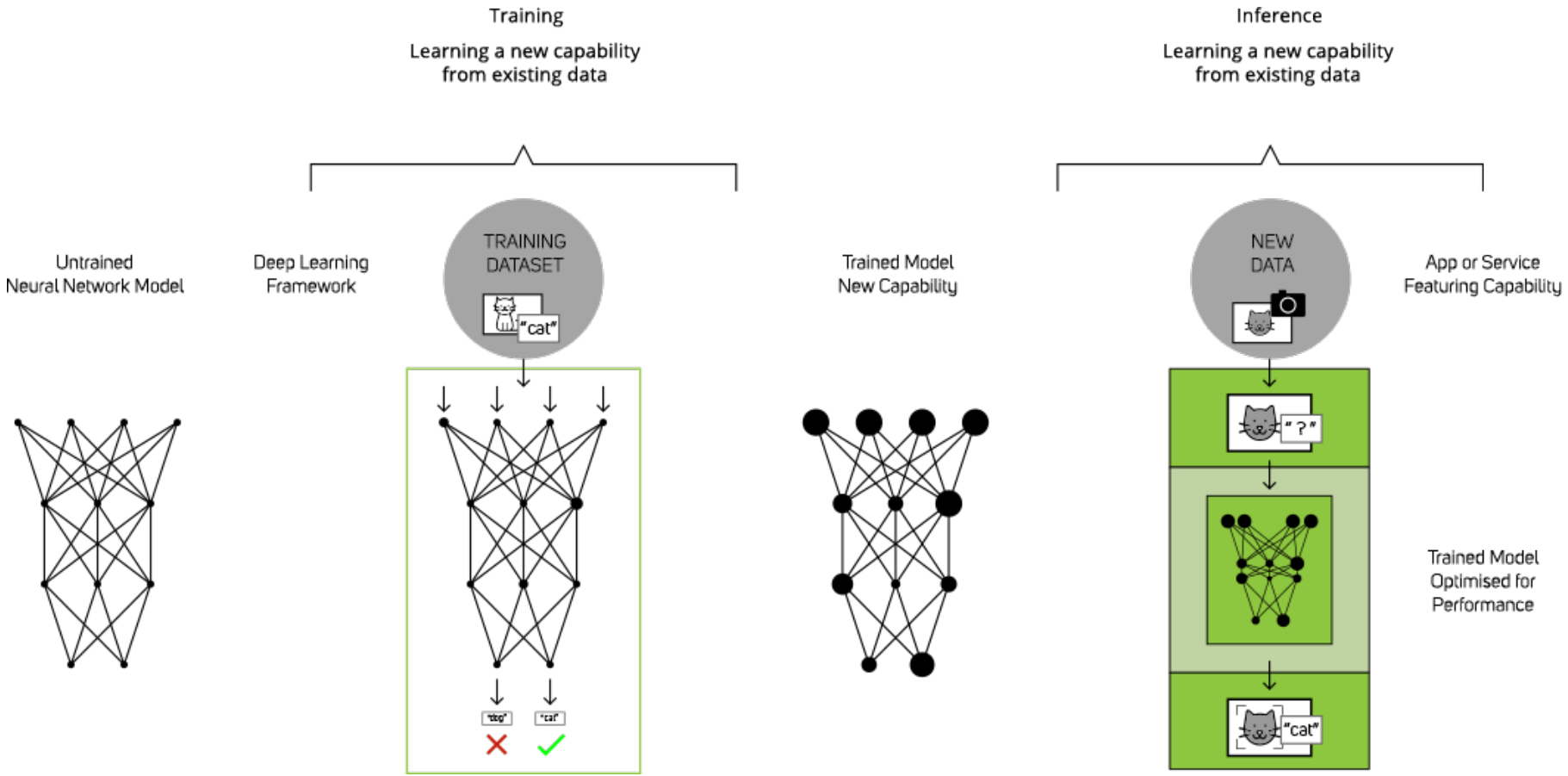
Personnage	Vecteur
	[-1.0, 1.0]
	[1.0, 0.0]
	[-1.0, 0.8]

La recherche vectorielle

classe les résultats par similarité (~pertinence)



Rank	Résultat
Requête	
1	
2	
3	
4	
5	



https://www.scan.co.uk/business/deep-learning-guide

elastic

Choisir son modèle d'Embedding

Commencer avec des modèles sur étagère

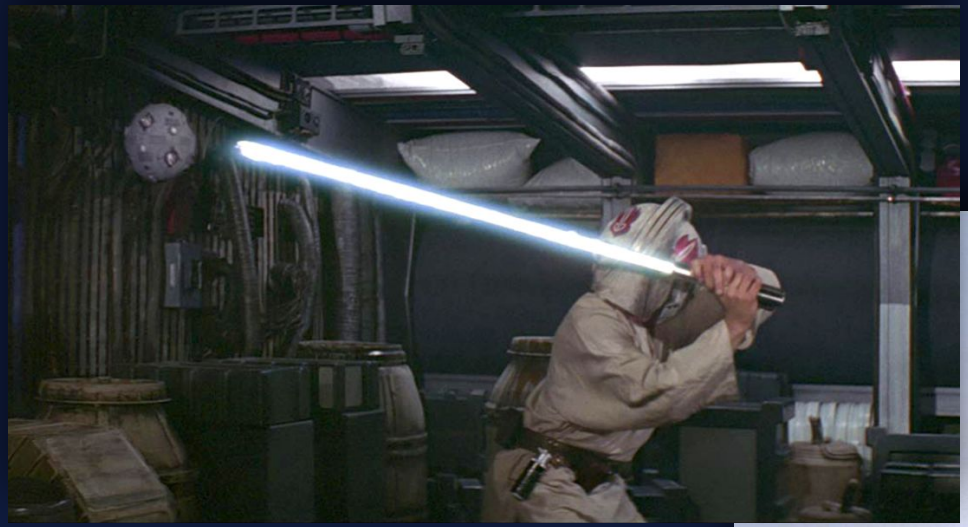
- Text data: Hugging Face (comme Microsoft E5)
- Images: OpenAI's CLIP

Développer pour une plus forte pertinence

- Appliquer un scoring hybride
- Bring Your Own Model : nécessite de l'expertise + des données labelisées

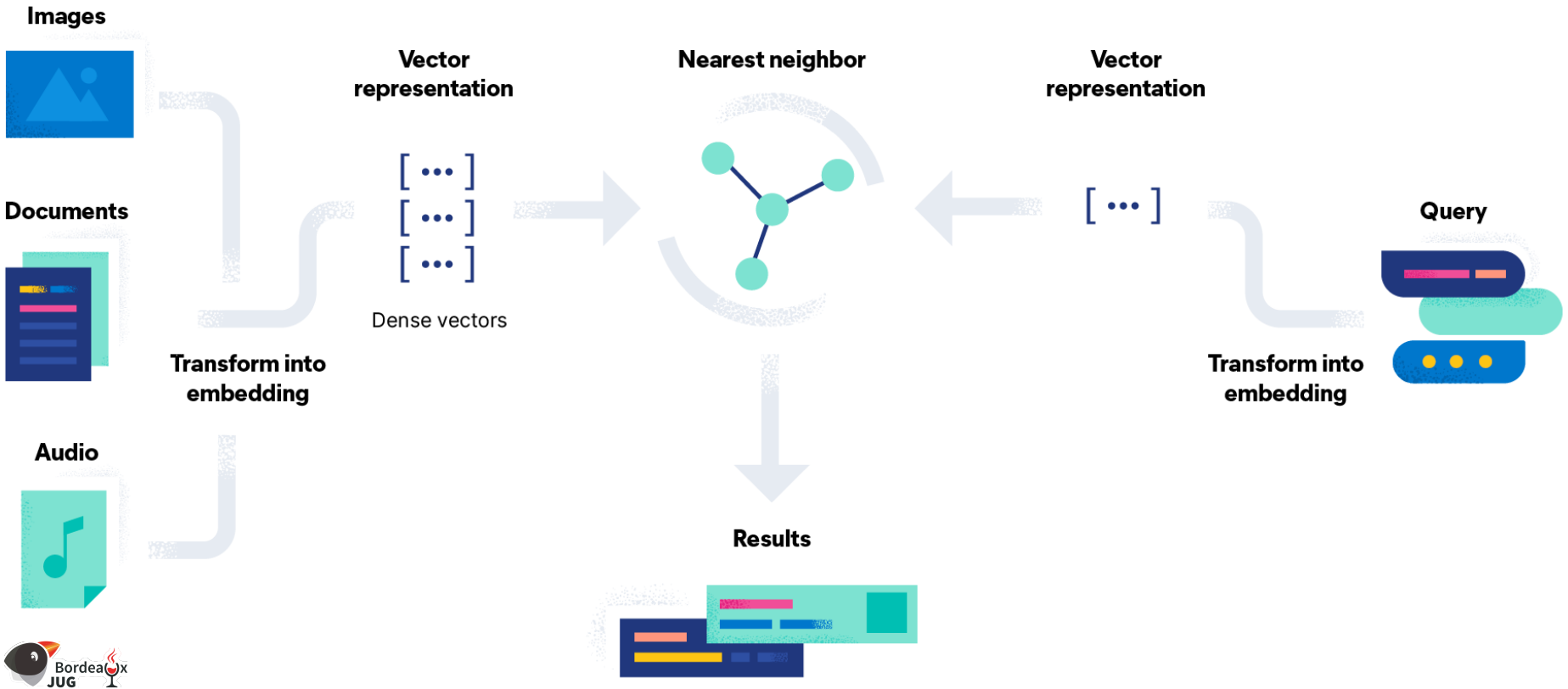
Problème

entraînement vs cas d'utilisation réel



elastic

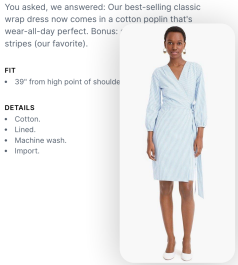
Recherche vectorielle



Comment indexer

avec des **vecteurs** ?

Ingestion des données avec embeddings

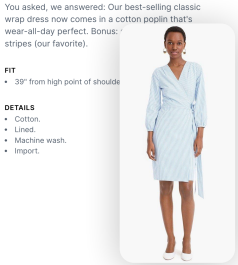


Source data

POST /_doc

```
{
  "_id":"product-1234",
  "product_name":"Summer Dress",
  "description":"Our best-selling…",
  "Price": 118,
  "color":"blue",
  "fabric":"cotton"
}
```

Ingestion des données avec embeddings



Source data

PyTorch

python™

POST /_doc

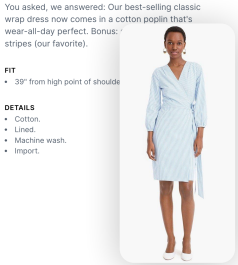
```
{
  "_id":"product-1234",
  "product_name":"Summer Dress",
  "description":"Our best-selling…",
  "Price": 118,
  "color":"blue",
  "fabric":"cotton",
  "desc_embedding":[0.452,0.3242,…],
  "img_embedding":[0.012,0.0,…]
}
```

Bordeaux JUG

elastic

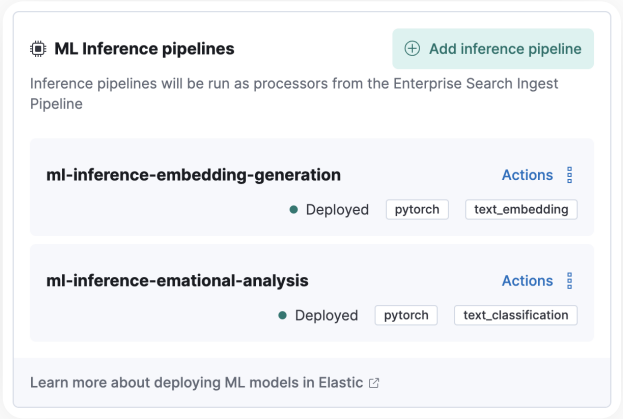
Avec Elastic ML

Commercial



Source data

```
POST /_doc
```

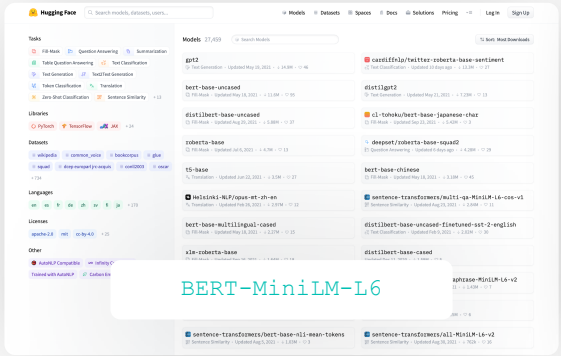


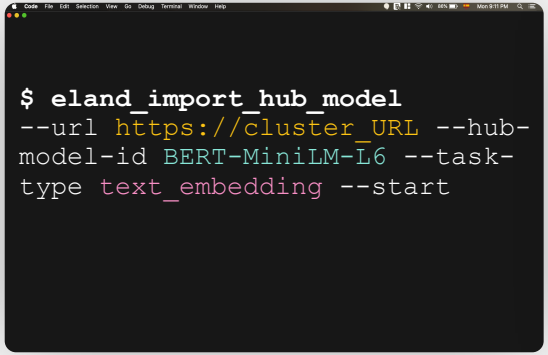

```
{
  "_id":"product-1234",
  "product_name":"Summer Dress",
  "description":"Our best-selling…",
  "Price": 118,
  "color":"blue",
  "fabric":"cotton",
  "desc_embedding":[0.452,0.3242,…]
}
```

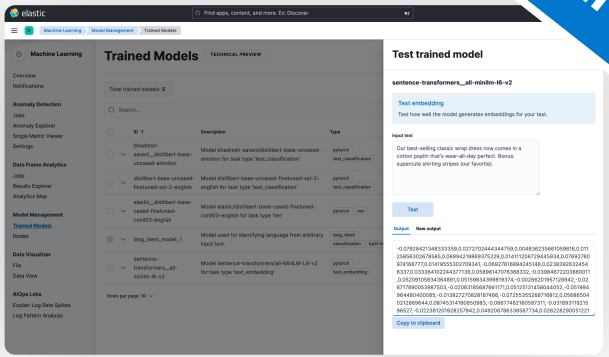
Bordeaux JUG

elastic

Configurer son modèle







PyTorch → eland →

Choisir le modèle approprié

Charger le modèle dans le cluster

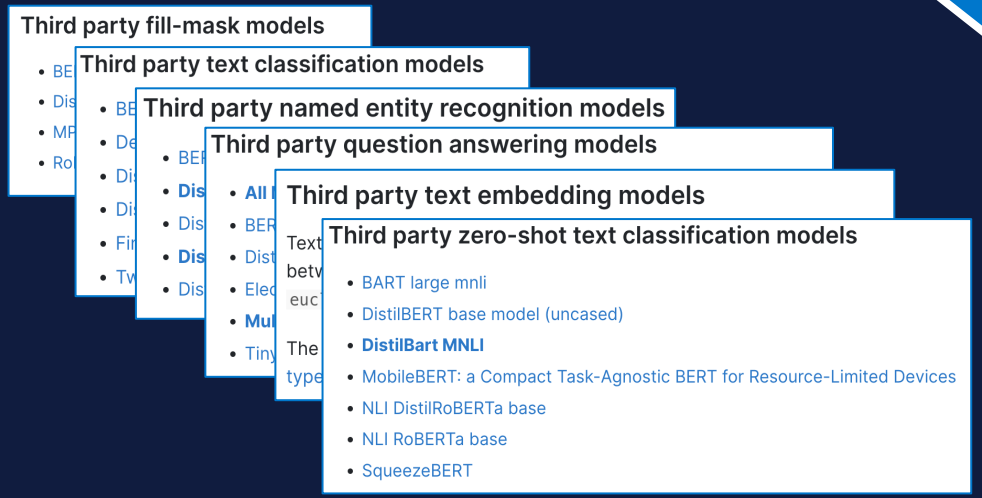
Gérer les modèles

Commercial

Gestion des modèles

Commercial

- C’est un domaine qui évolue rapidement. La flexibilité vous permet de vous adapter facilement.
- Utilisation des modèles tiers PyTorch
- Support de plusieurs types de modèles NLP



Liste complète des modèles supportés par Elastic : ela.st/nlp-supported-models

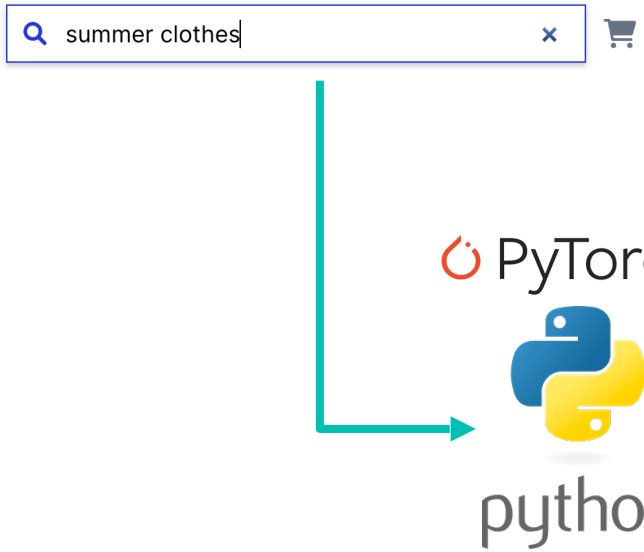
Bordeaux JUG

elastic

Comment chercher

avec des **vecteurs** ?

Requête Vectorielle




```
GET product-catalog/_search
{
  "knn": {
    "field": "desc_embbeding",
    "k": 5,
    "num_candidates": 50,
    "query_vector": [0.123, 0.244,...],
    "filter": {
      "term": {
        "department": "women"
      }
    }
  },
  "size": 10
}
```

Requête Vectorielle

Transformer model

PyTorch

```
GET product-catalog/_search
{
  "knn": {
    "field": "desc_embbeding",
    "k": 5,
    "num_candidates": 50,
    "query_vector_builder": {
       "text_embedding": {
         "model_text": "summer clothes",
         "model_id": <text-embedding-model>
    },

    "filter": {
       "term": {
         "department": "women"
       }
    }
  },
  "size": 10
}
```

Les composants de la recherche vectorielle

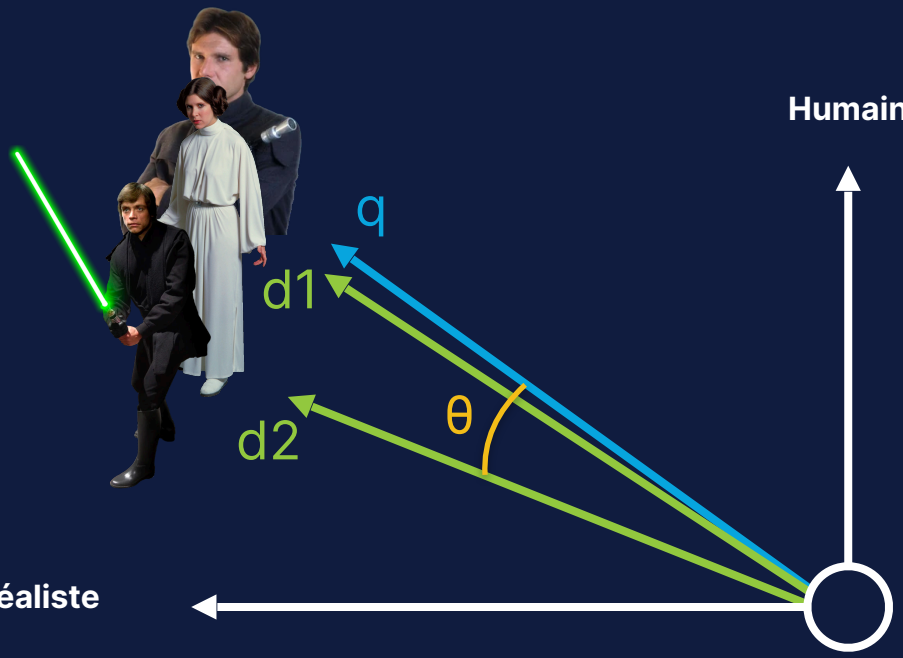
Recherche	Indexation	Génération
Query	**Mapping**	**Embedding**
`kNN`	`dense_vector`	Text embedding model (3rd party, local, in Elasticsearch)

elastic

Mais comment ça fonctionne

en vrai ?

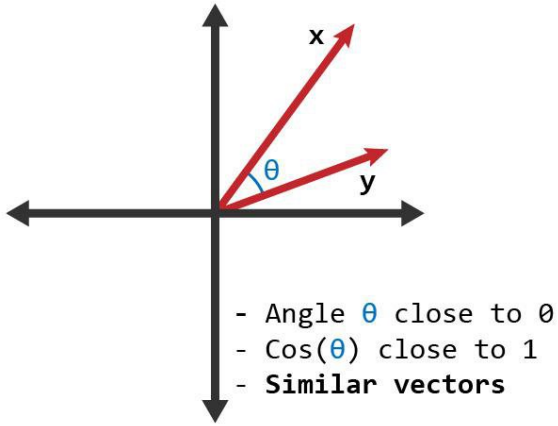
Similarité : cosinus (cosine)



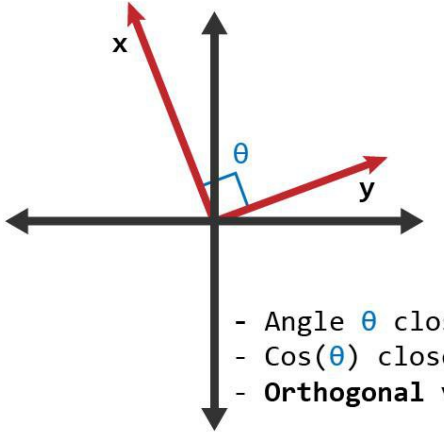
$$cos(\theta) = \frac{\vec{q} \times \vec{d}}{|\vec{q}| \times |\vec{d}|}$$

$$\_score = \frac{1 + cos(\theta)}{2}$$

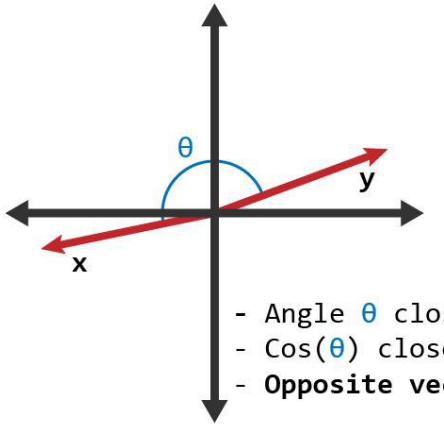
Similarité : cosinus (cosine)
rappel



- Angle θ close to 0
- Cos(θ) close to 1
- **Similar vectors**



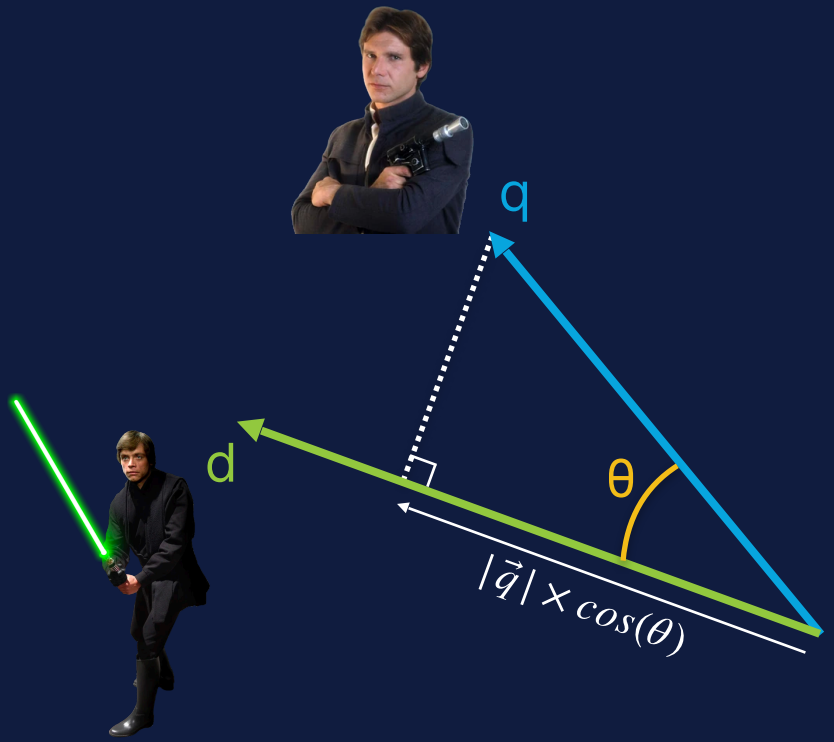
- Angle θ close to 90
- Cos(θ) close to 0
- **Orthogonal vectors**



- Angle θ close to 180
- Cos(θ) close to -1
- **Opposite vectors**

elastic

Similarité : produit scalaire (dot_product)



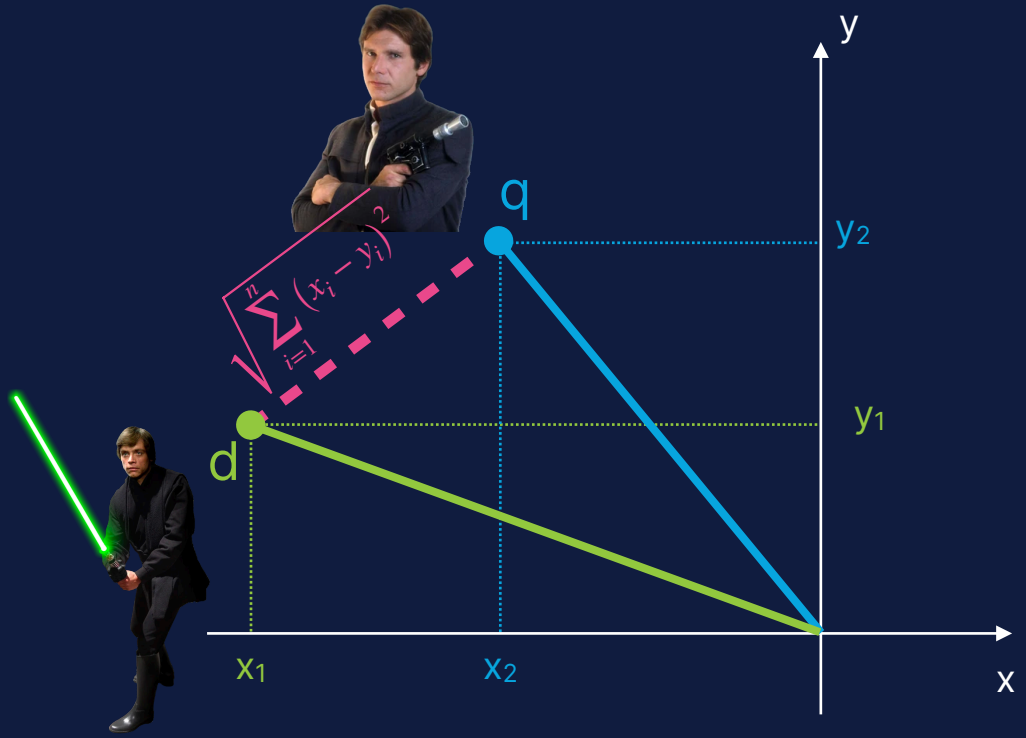
$$\vec{q} \times \vec{d} = |\vec{q}| \times cos(\theta) \times |\vec{d}|$$

$$\_score_{float} = \frac{1 + dot\_product(q, d)}{2}$$

$$\_score_{byte} = \frac{0.5 + dot\_product(q, d)}{32768 \times dims}$$

elastic

Similarité : distance euclidienne (l2_norm)



$$l2\_norm_{q,d} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

$$\_score = \frac{1}{1 + (l2\_norm_{q,d})^2}$$

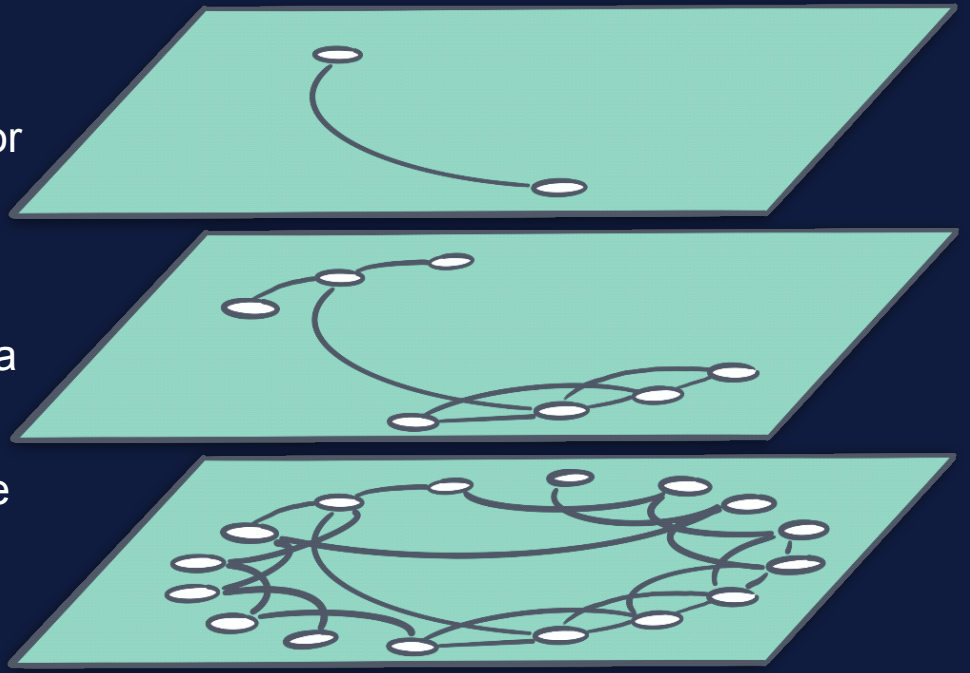
elastic

Brute Force

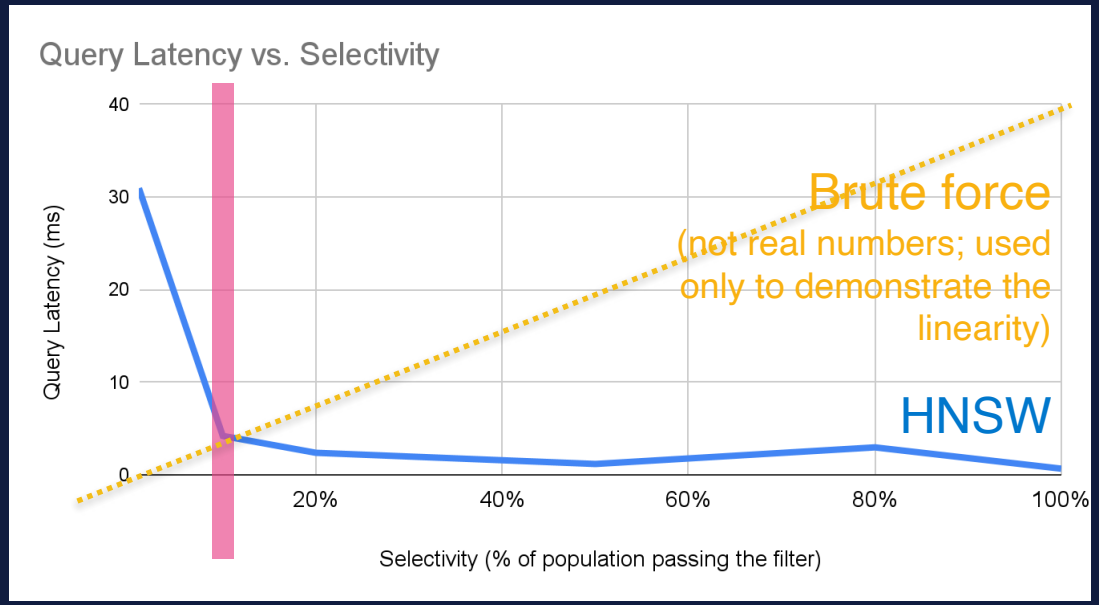
elastic

Hierarchical Navigable Small Worlds (HNSW)

- **HNSW:** a layered approach that simplifies access to the nearest neighbor
- **Tiered:** from coarse to fine approximation over a few steps
- **Balance:** Bartering a little accuracy for a lot of scalability
- **Speed:** Excellent query latency on large scale indices



brute force ou HNSW ?

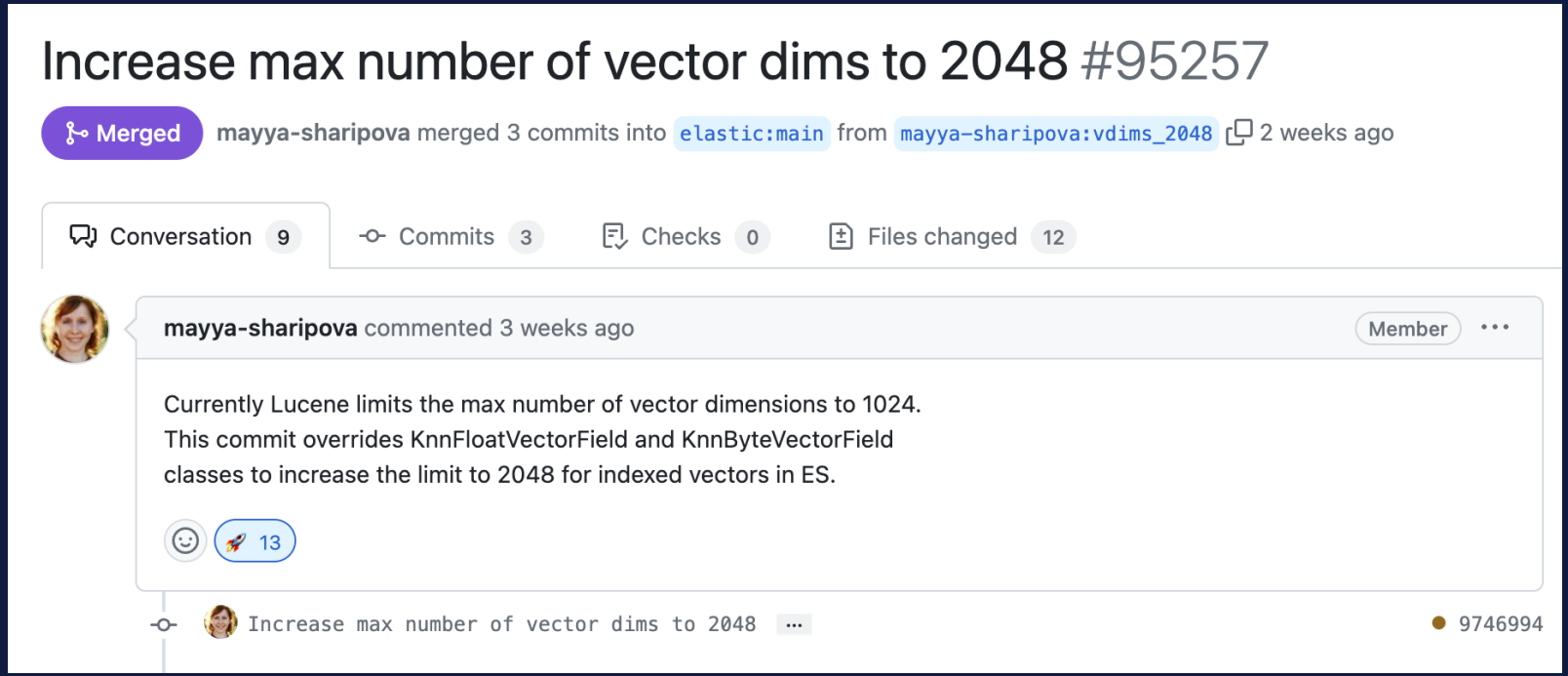


Au pire : 2*(brute force)

- Brute force : O(n) des documents filtrés
- HNSW : ~O(log(n)) de tous les documents

elastic

Elasticsearch + Lucene
des progrès rapides ❤️



“Scaler” la recherche vectorielle

Recherche vectorielle

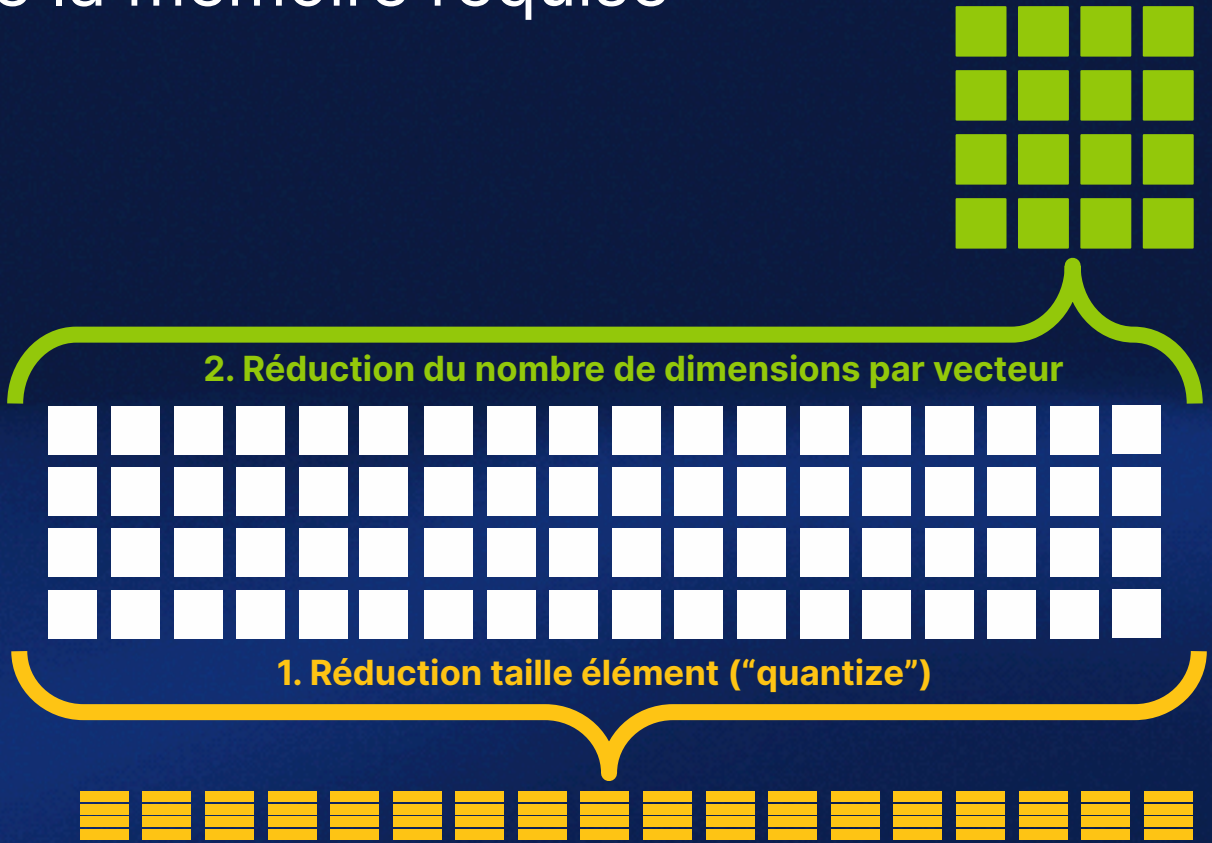
1. Besoin énorme de mémoire
2. Indexation plus lente
3. Merge est plus lent

* Améliorations permanentes dans Lucene et Elasticsearch

Bonnes pratiques

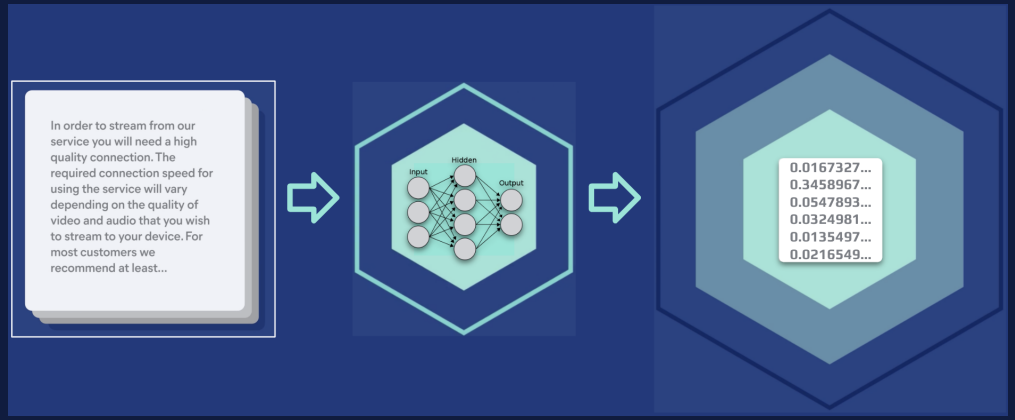
1. Eviter les recherches pendant l’indexation
2. Exclure les vecteurs du champ _source
3. Réduire le nombre de dimensions des vecteurs
4. Utiliser des bytes plutôt que des float

Réduire la mémoire requise



Base de données vectorielles

- **Stockage efficace** de vecteurs numériques, support des opérations **CRUD**
- **Recherche rapide** de vecteurs
- Conçu pour de la recherche vectorielle à grande échelle (**scalable**)



Elasticsearch

bien plus qu'une base de données vectorielles

Elasticsearch est capable de :

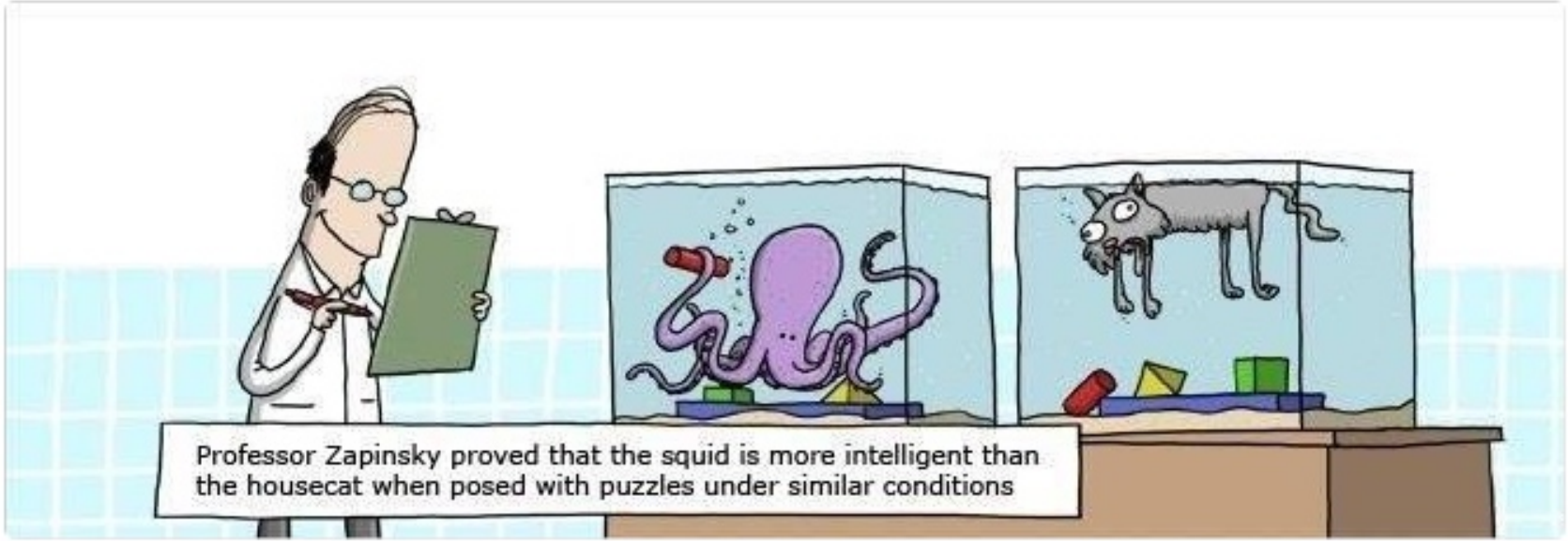
- **Créer des vector embeddings** (représentations numériques des données)
- Est **optimisé pour stocker** des vecteurs éparpillés et denses, en volume

Elasticsearch a aussi :

- **Filtres et agrégations :** sur l'ensemble des résultats
- **Sécurité native au niveau document* :** utilisé en production par les clients enterprise
- **Types de données :** Geo, full text, support des langues
- **Outils d'ingestion :** connecteurs, API, crawler web et des intégrations tierces
- Communité, adoption massive par les enterprises, Track record, Elastic stack...

* nécessite une license commerciale

Benchmarketing



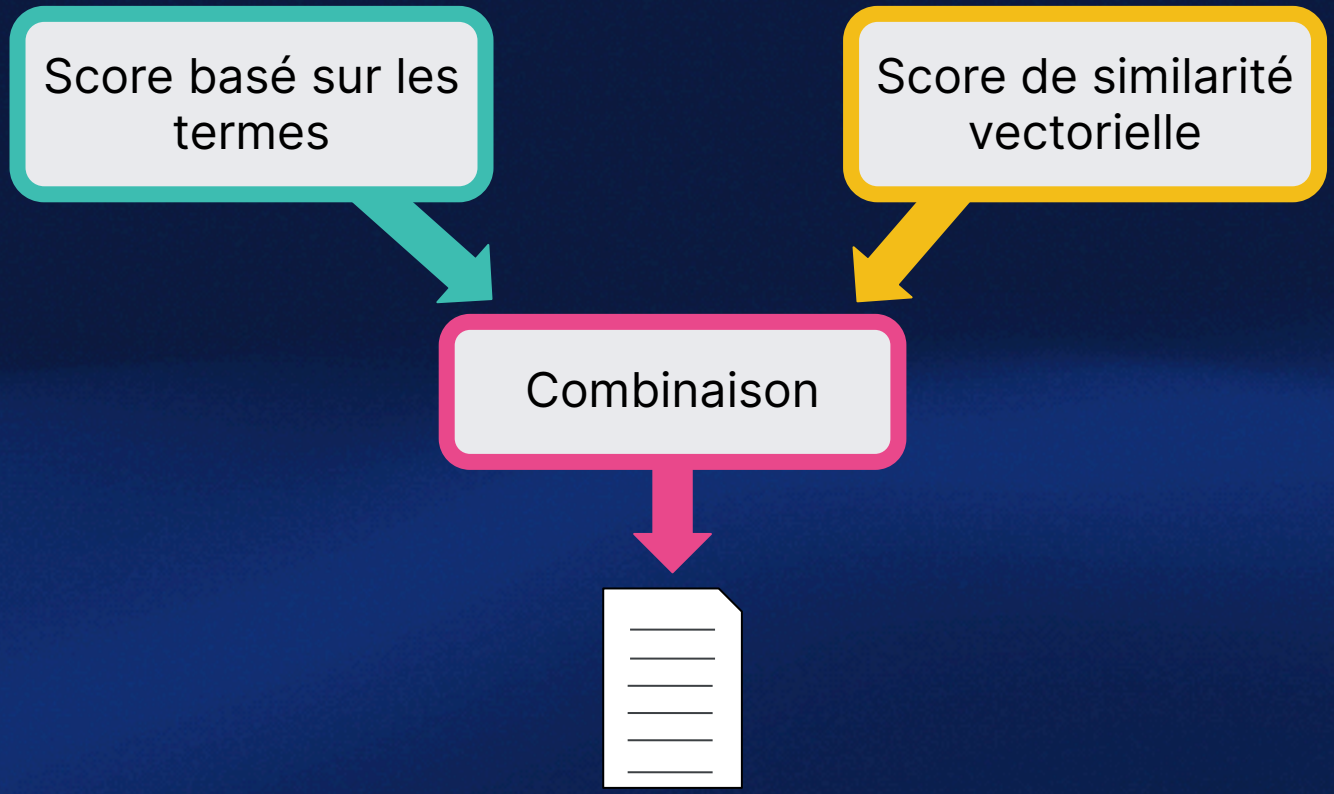
Elasticsearch

You Know, for **Hybrid** Search

Bordeaux JUG

elastic

Calcul hybride du score de pertinence



```
GET product-catalog/_search
{
  "query": {
    "match": {
      "description": {
        "query": "summer clothes",
        "boost": 0.9
      }
    }
  },
  "knn": {
    "field": "desc_embbeding",
    "query_vector": [0.123, 0.244,...],
    "k": 5,
    "num_candidates": 50,
    "boost": 0.1,
    "filter": {
       "term": {
         "department": "women"
       }
     }
   },
  "size": 10
}
```

```
GET product-catalog/_search
{
  "query": {
    "match": {
      "description": {
        "query": "summer clothes",
        "boost": 0.9
      }
    }
  },
  "knn": [ {
    "field": "image-vector",
    "query_vector": [54, 10, -2],
    "k": 5,
    "num_candidates": 50,
    "boost": 0.1
  },
  {
    "field": "title-vector",
    "query_vector": [1, 20, -52, 23, 10],
    "k": 10,
    "num_candidates": 10,
    "boost": 0.5
  }],
  "size": 10 }
```

ELSER

Elastic Learned Sparse EncodER

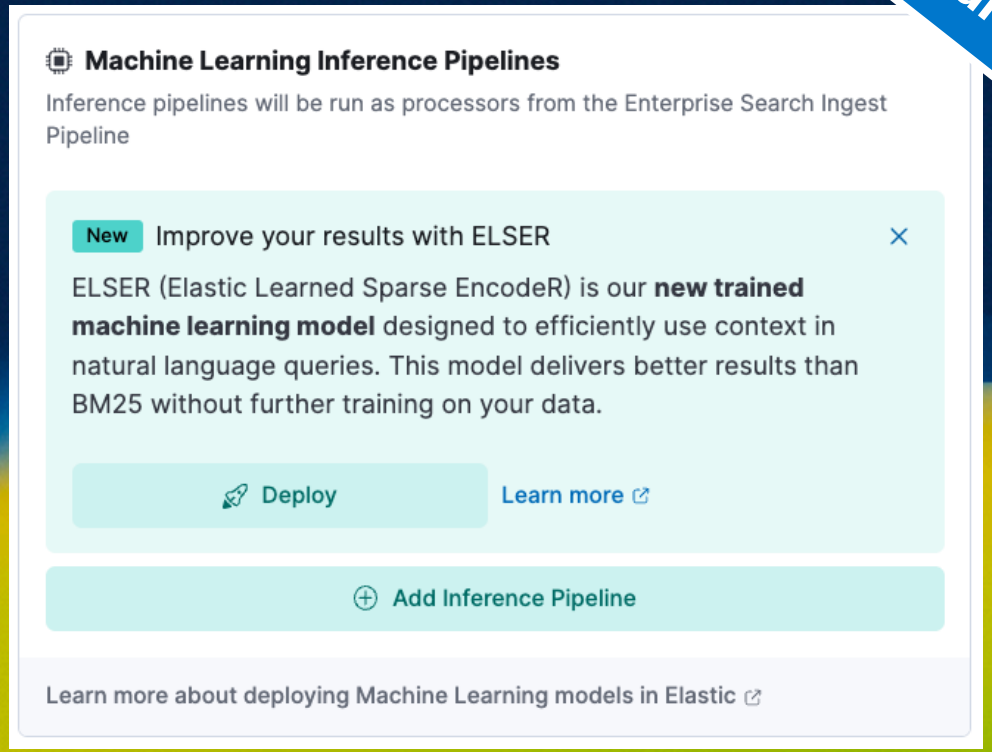
Commercial

text_expansion

Not BM25 or (dense) vector

Sparse vector like BM25

Stored as inverted index



Bordeaux JUG

elastic

ELSER - pertinence sur étagère, multi domaines

Commercial

- Permet d'implémenter la recherche sémantique sans avoir à entraîner votre propre modèle
- Généraliste : plein de domaines sans entraînement
- Excellente pertinence, sur étagère

Search results ranking models ↓ — Data sets (BEIR benchmark) →

Démo

	Average	TREC-COVID	NFCorpus	NQ	HotpotQA	FiQA	ArguAna	Touche-2020	DBPedia	SCIDOCS	FEVER	Climate-FEVER	SciFact
BM25	0.416	0.688	0.327	0.326	0.602	0.254	0.472	0.347	0.287	0.165	0.649	0.186	0.69
RRF (BM25/Dense)	0.449	0.697	0.317	0.445	0.611	0.318	0.474	**0.354**	0.353	0.159	0.746	0.238	0.671
Linear (BM25/Dense)	0.471	0.787	0.335	0.485	0.62	0.341	0.444	0.346	0.378	0.164	**0.778**	**0.272**	0.698
SPLADE	N/A	0.726	0.347	**0.537**	0.687	**0.347**	**0.526**	0.246	**0.436**	0.158			0.703
Elastic Learned Sparse Encoder	0.471	0.747	**0.351**	0.524	0.67	0.339	0.5	0.263	0.415	0.156	0.777	0.218	0.695
RRF (BM25 / Elastic Learned Sparse Encoder)	**0.478**	**0.797**	**0.352**	0.468	0.674	0.311	0.497	0.347	0.411	**0.166**	0.762	0.24	**0.712**

Search quality *worse* than benchmark (BM25) — Search quality *better* than benchmark (BM25)

Bordeaux JUG

elastic

```
POST /_ingest/pipeline/_simulate
{
  "pipeline":{"processors":[
    {
      "inference":{"model_id":".elser_model_1"}
    }
  ]},
  "docs": [
    {"_source":{"text_field":"These are not the droids you
   are looking for."}},
    {"_source":{"text_field":"Obi-Wan never told you what
   happened to your father."}},
    {"_source":{"text_field": "No. I am your father!"}}
  ]
}
```

Commercial

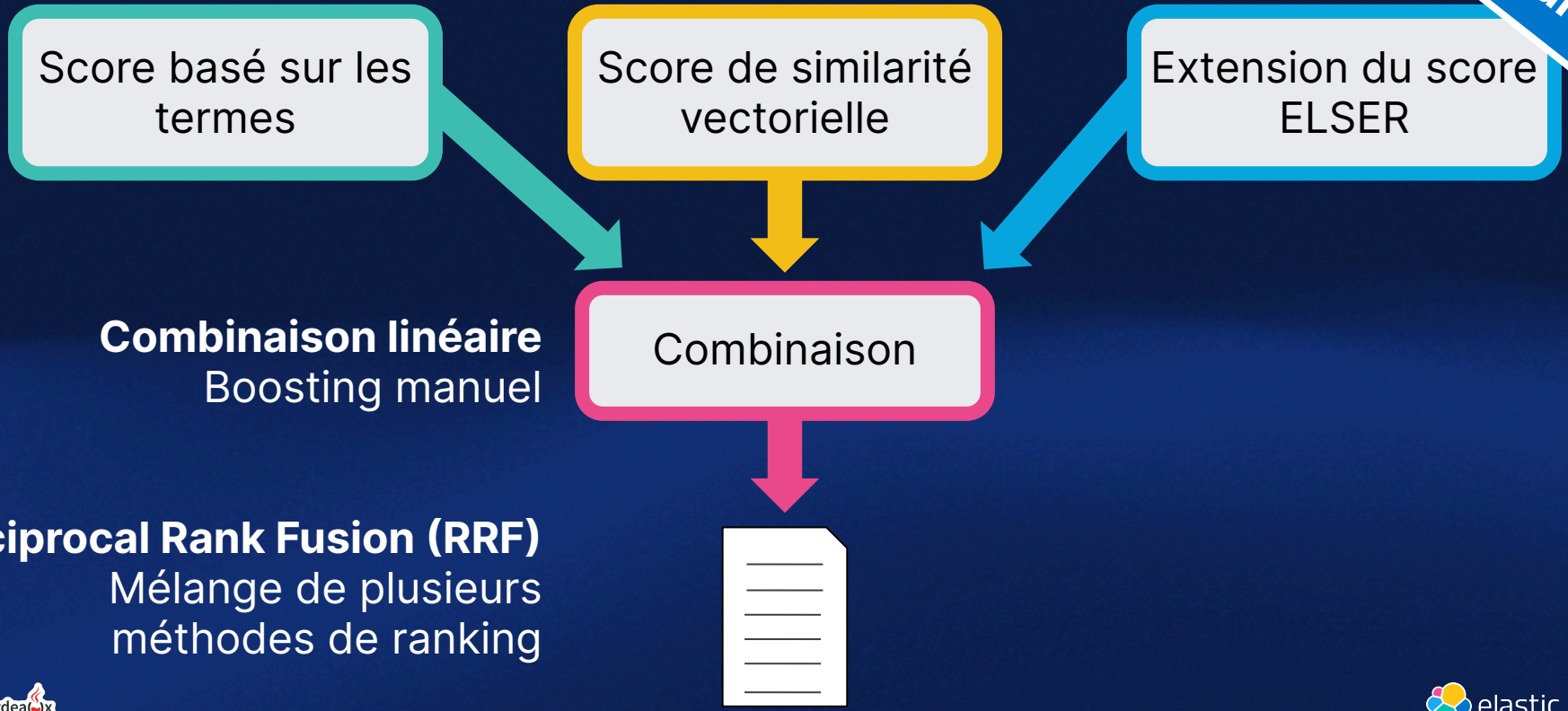
Commercial

These are not the droids you are looking for.

```
"ml": {
        "inference": {
          "predicted_value": {
            "lucas": 0.50047517,
            "ship": 0.29860738,
            "dragon": 0.5300422,
            "quest": 0.5974301,
            "dr": 2.1055143,
            "space": 0.49377063,
            "robot": 0.40398192,
            ...
```

Combinaison de score

Commercial



Combinaison linéaire
Boosting manuel

Reciprocal Rank Fusion (RRF)
Mélange de plusieurs méthodes de ranking

Reciprocal Rank Fusion (RRF)

$$RRFscore(d \in D) = \sum_{r \in R} \frac{1}{k + r(d)}$$

D - set of docs
R - set of rankings as permutation on 1..|D|
k - typically set to 60 by default

Ranking Algorithm 1			
Doc	Score	r(d)	k+r(d)
A	1	1	61
B	0.7	2	62
C	0.5	3	63
D	0.2	4	64
E	0.01	5	65

Ranking Algorithm 2			
Doc	Score	r(d)	k+r(d)
C	1,341	1	61
A	739	2	62
F	732	3	63
G	192	4	64
H	183	5	65

Doc	RRF Score
A	1/61 + 1/62 = 0,0325
C	1/63 + 1/61 = 0,0323
B	1/62 = 0,0161
F	1/63 = 0,0159
D	1/64 = 0,0156

```
GET product-catalog/_search
{
  "sub_searches": [
    {
      "query": {
        "match": {...}
      }
    },
    {
      "query": {
        "text_expansion": {...}
      }
    }
  ],
  "knn": {...},
  "rank": {
    "rrf": {
      "window_size": 50,
      "rank_constant": 20
    }
  }
}
```

BM25f

+

ELSER

+

Vector

Recherche Hybride avec RRF

Commercial

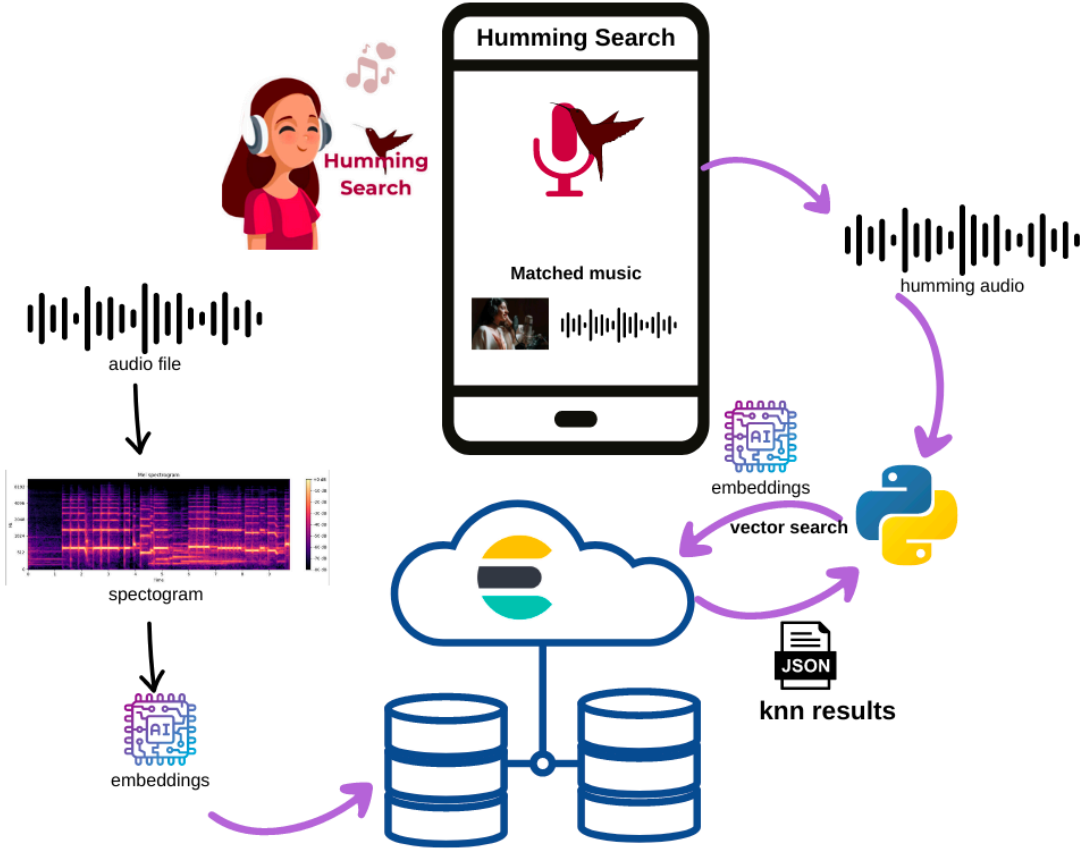
Gratuit vs **Payant** (platinum)

kNN with HNSW

1. Fournissez vos propres embeddings
2. Attention à la mémoire requise pour faire tenir les vecteurs en RAM (off-heap)

Inference in Elasticsearch

1. Génération des embeddings (transformers HuggingFace, autres)
2. ELSER



https://github.com/dadoonet/music-search/

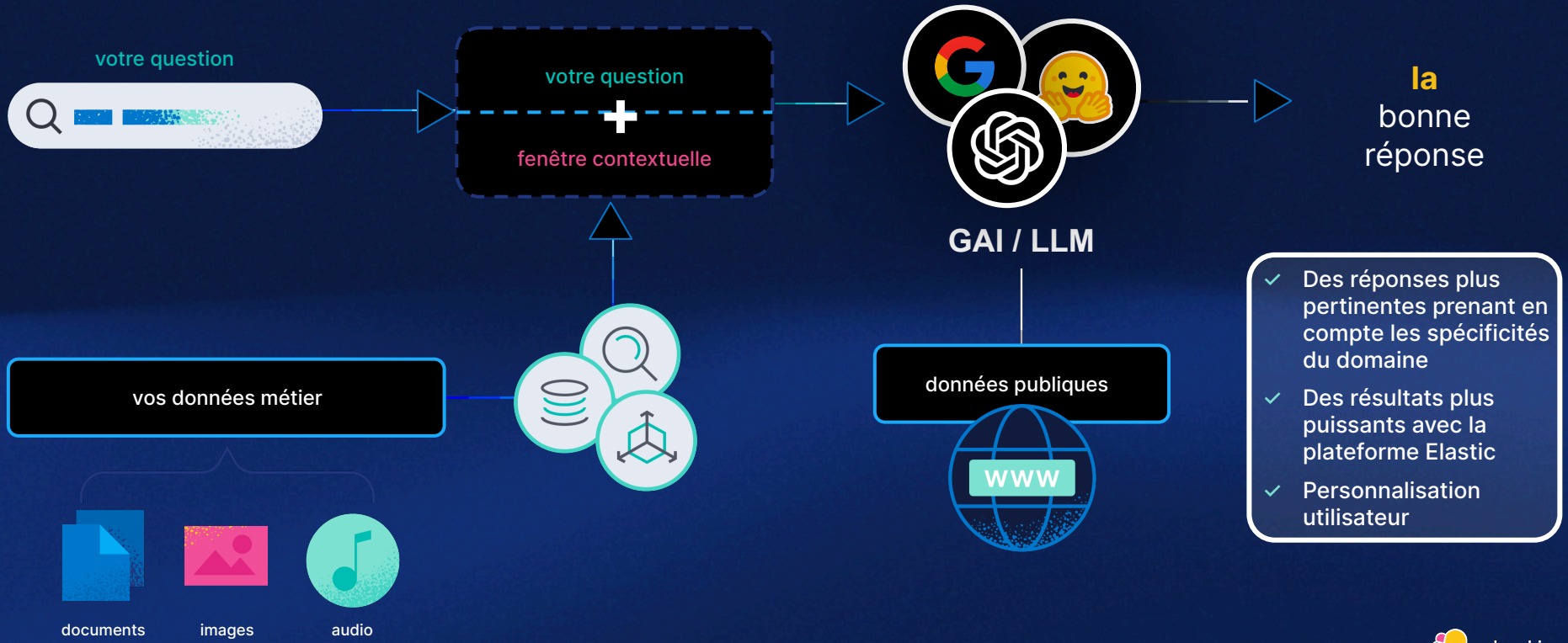
ChatGPT

Elastic et les LLMs

LLMs : opportunités et limitations

elastic

Retrieval Augmented Generation

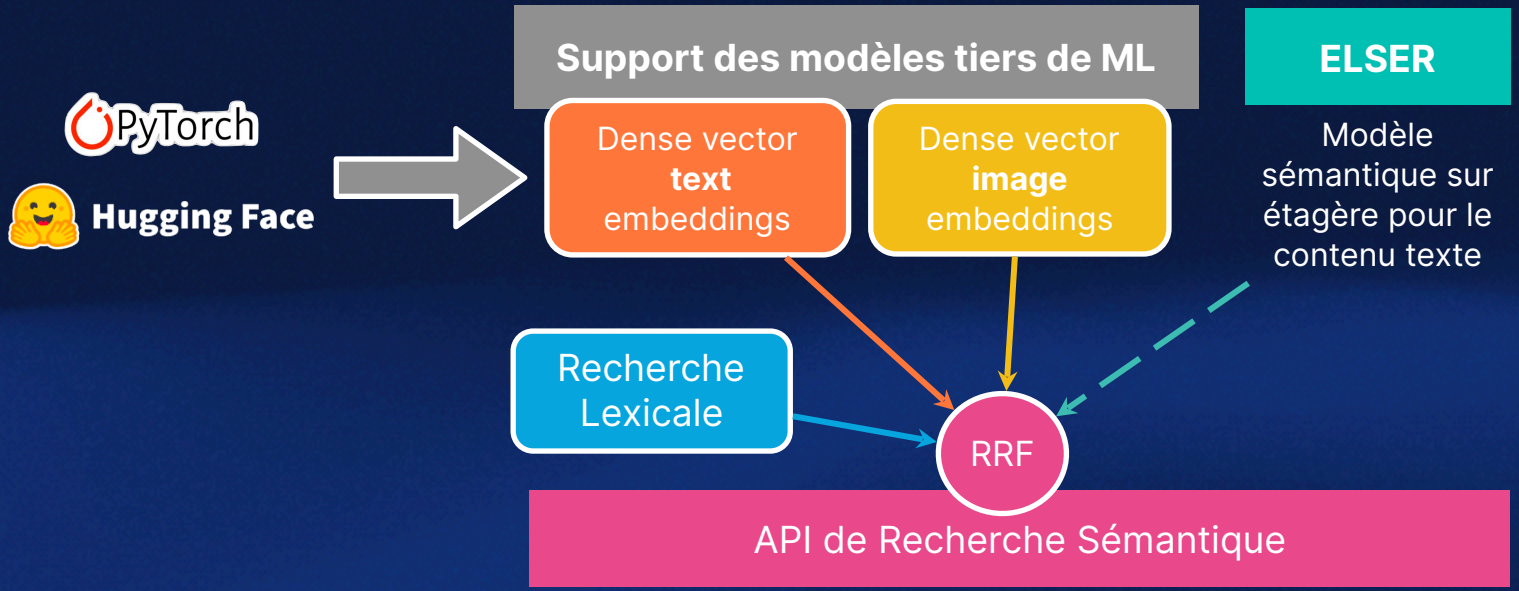


elastic

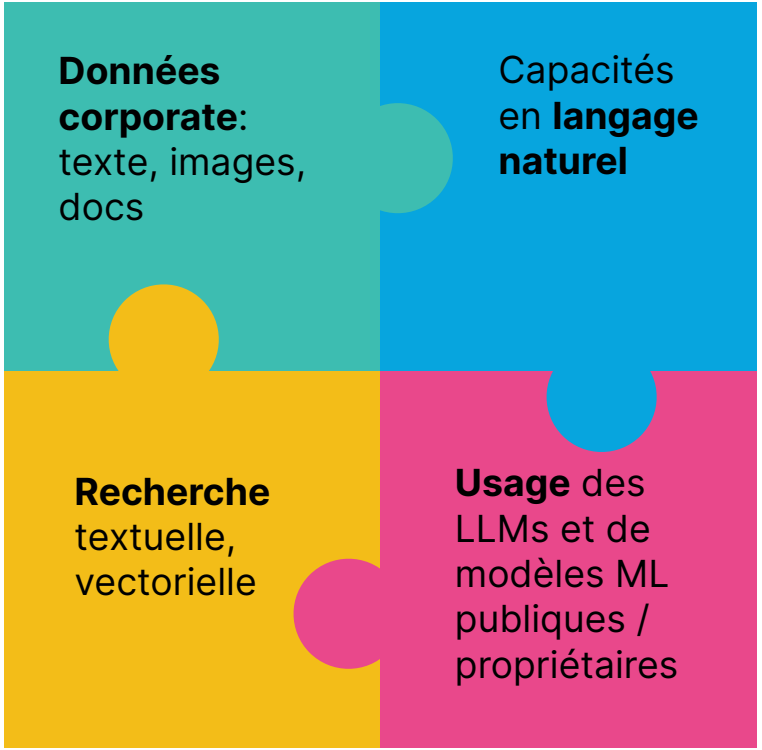
Démo

Conclusion

La recherche lexicale enrichie en sémantique



Les ingrédients nécessaires pour la recherche IA...



Bordeaux JUG

elastic

… sont les fonctionnalités que nous livrons en standard

Bordeaux JUG

elastic

La **recherche** à l'ère de l'**IA**

David Pilato | @dadoonet