elastic

Search & AI:
a new era

David Pilato | @dadoonet

LYON JUG

Agenda

Commercial

- "Classic" search and its limitations
- ML model and usage
- Vector search or hybrid search in Elasticsearch
- OpenAI's ChatGPT or LLMs with Elasticsearch

elastic

Elasticsearch

You Know, for Search

elastic

Elasticsearch

Lucene

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."
}
```

elastic

These are not 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
  }]}
```

Semantic
search
≠
Literal
matches

TODAY

X-wing starfighter squadron

TOMORROW

What ships and crews do I need to destroy an almost finished death star?
Or is there a secret weakness?

Elasticsearch

You Know, for Search

elastic

Elasticsearch

You Know, for **Vector** Search

elastic

What is a Vector?

elastic

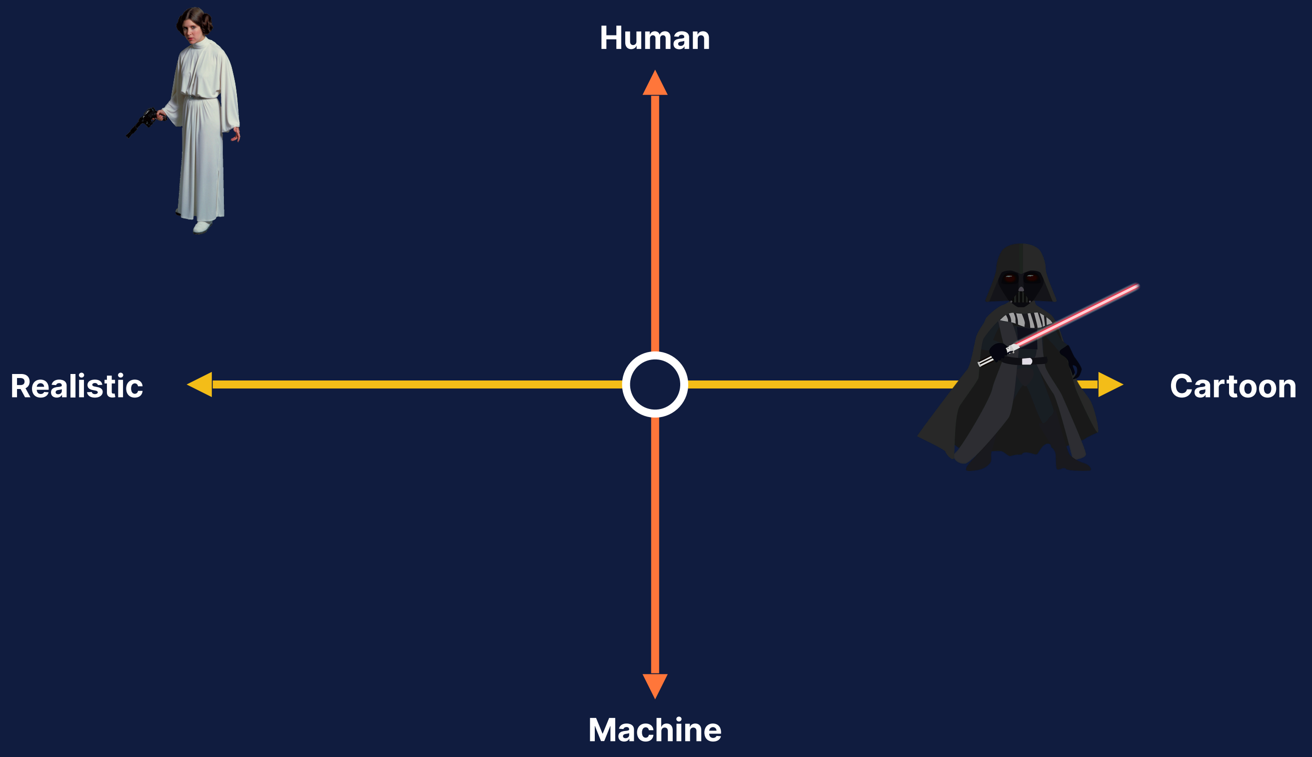
Embeddings represent your data

Example: 1-dimensional vector

Character	Vector
	[-1]
	[1]

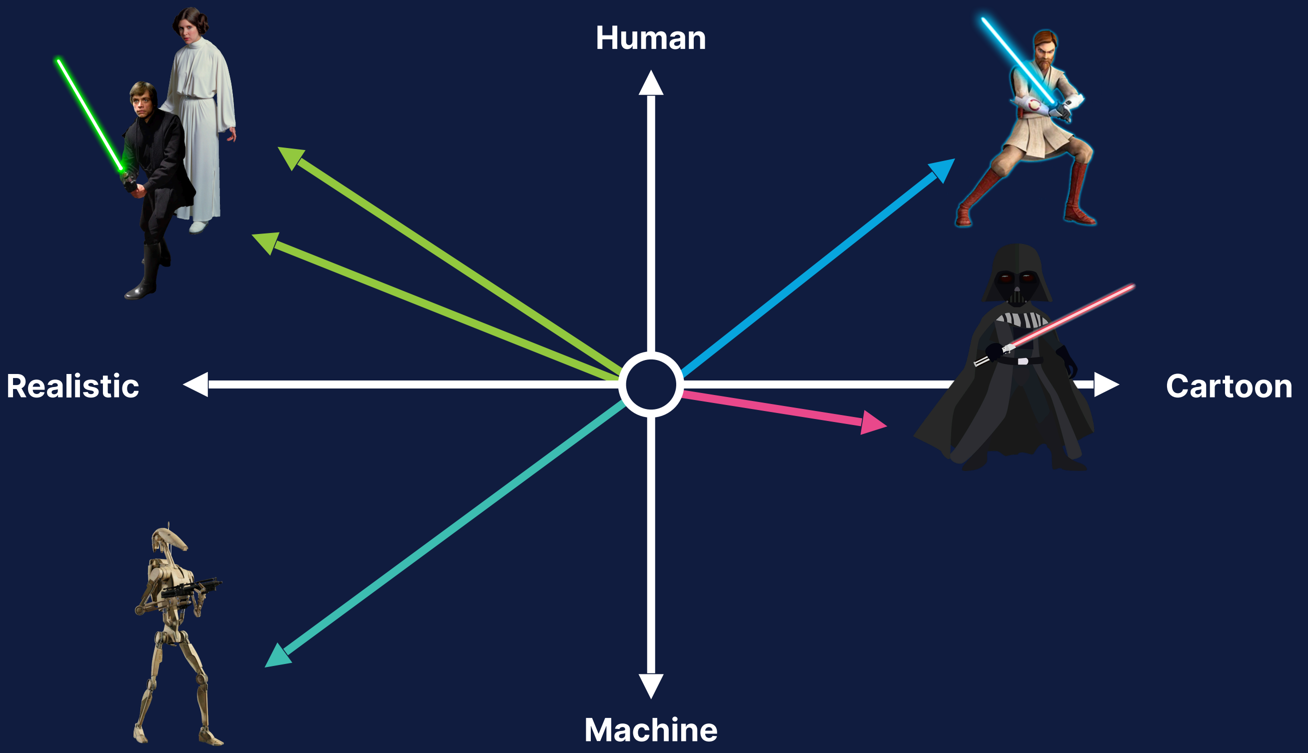
elastic

Multiple dimensions
represent different data aspects



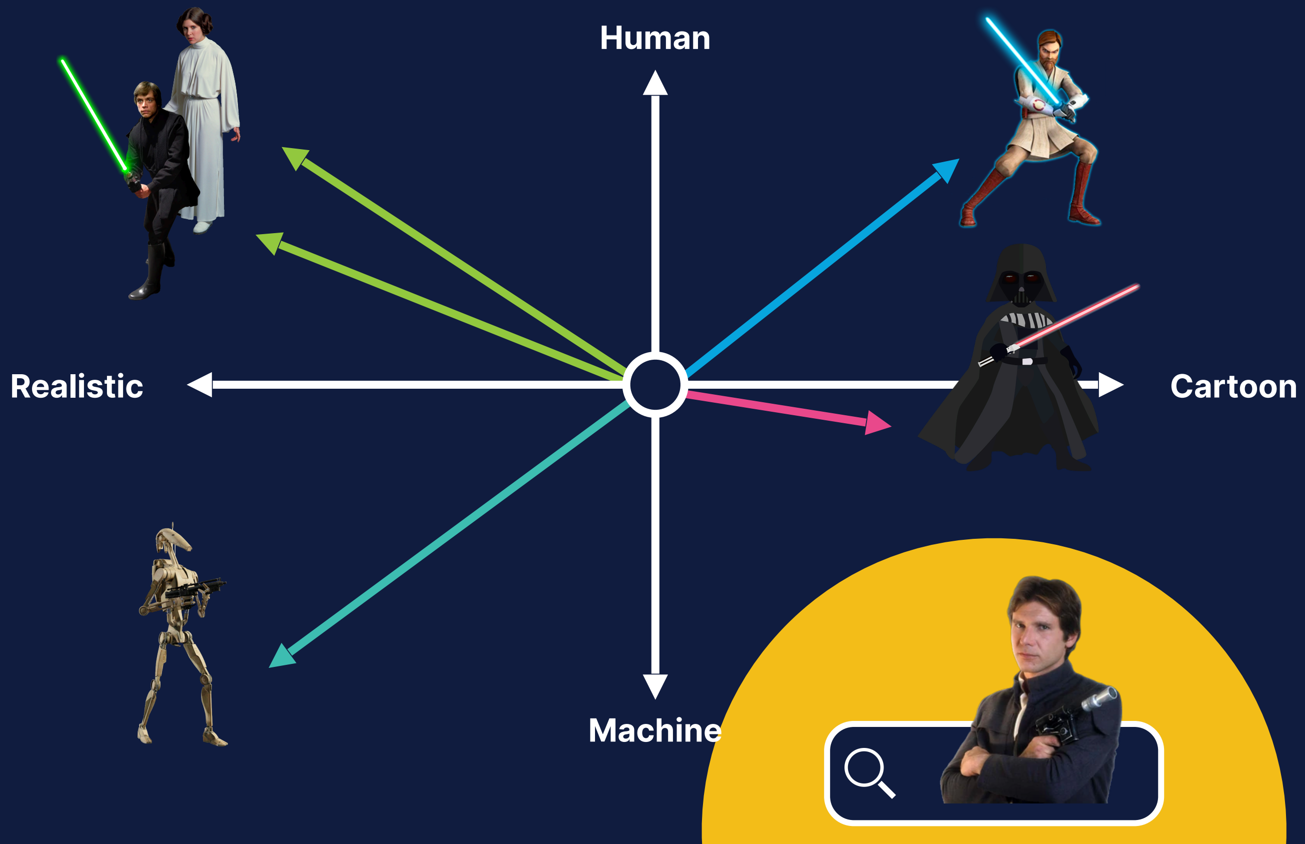
Character	Vector
	[-1, 1]
	[1, 0]

Similar data
is grouped together



Character	Vector
	[-1.0, 1.0]
	[1.0, 0.0]
	[-1.0, 0.8]

Vector search ranks objects
by similarity (~relevance) to the query



Rank	Result
Query	
1	
2	
3	
4	
5	

Choice of Embedding Model

Start with Off-the Shelf Models

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

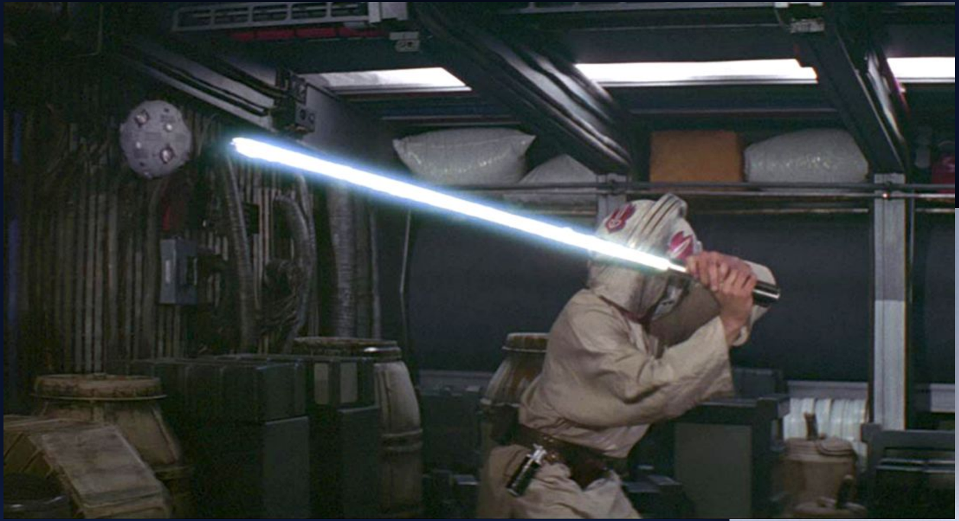
Extend to Higher Relevance

- Apply hybrid scoring
- Bring Your Own Model: requires expertise + labeled data

elastic

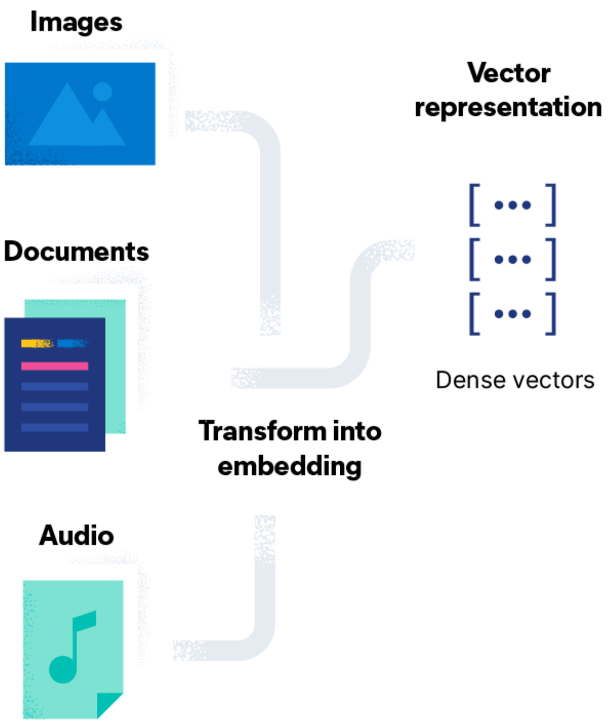
Problem

training vs actual use-case



elastic

Architecture of Vector Search

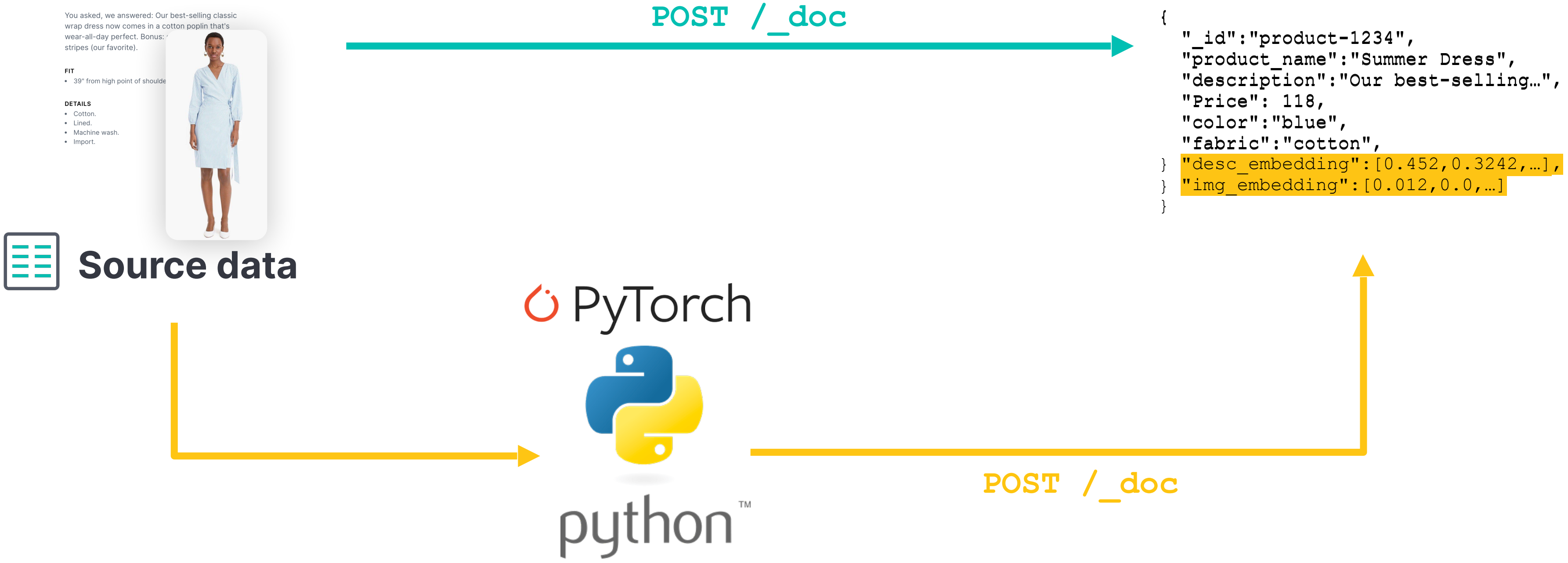


How do you

index vectors?

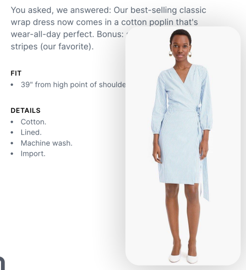
elastic

Data Ingestion and Embedding Generation



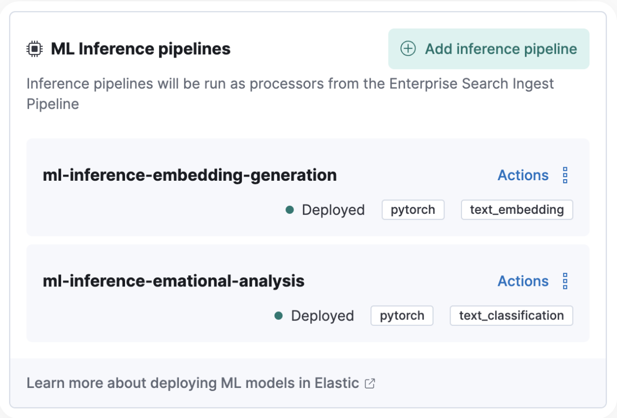
With Elastic ML

Commercial



Source data

POST /_doc

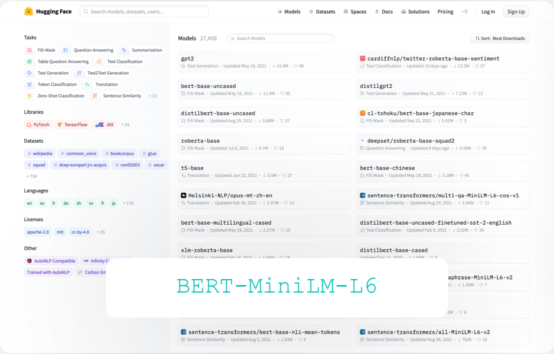


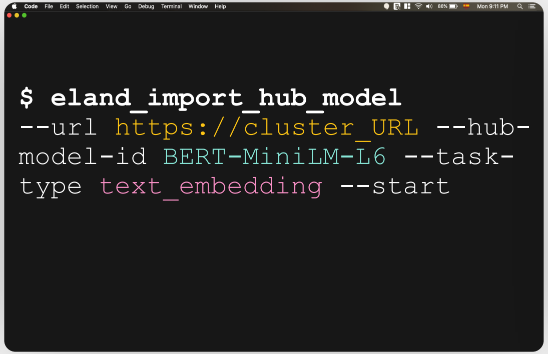

```
{
  "_id":"product-1234",
  "product_name":"Summer Dress",
  "description":"Our best-selling…",
  "Price": 118,
  "color":"blue",
  "fabric":"cotton",
  "desc_embedding":[0.452,0.3242,…]
}
```

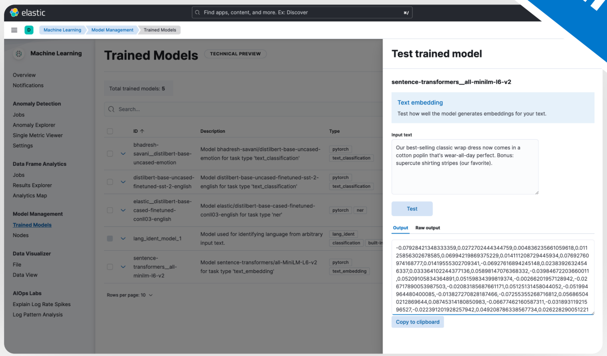
elastic

Commercial

Eland Imports PyTorch Models







PyTorch → eland →

Select the appropriate model

Load it

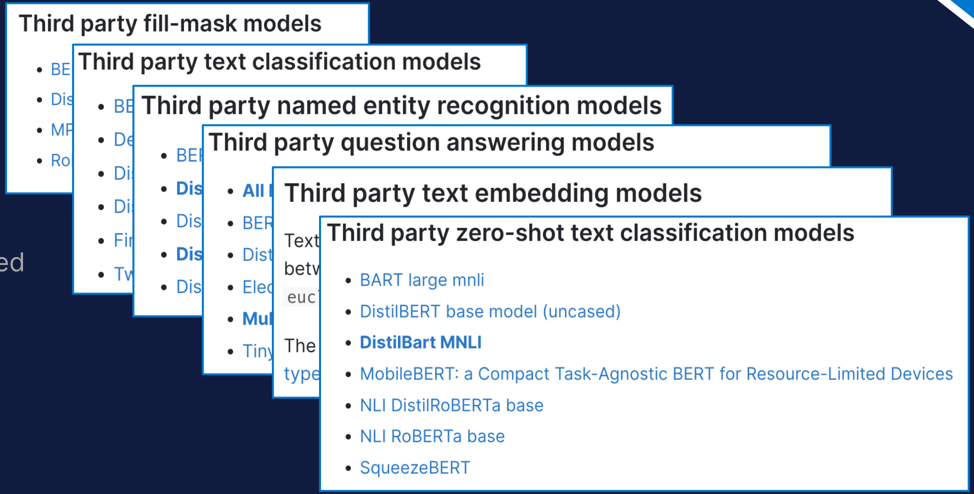
Manage models

elastic

Elastic's range of supported NLP models

Commercial

- **Fill mask model**
  Mask some of the words in a sentence and predict words that replace masks
- **Named entity recognition model**
  NLP method that extracts information from text
- **Text embedding model**
  Represent individual words as numerical vectors in a predefined vector space
- **Text classification model**
  Assign a set of predefined categories to open-ended text
- **Question answering model**
  Model that can answer questions given some or no context
- **Zero-shot text classification model**
  Model trained on a set of labeled examples, that is able to classify previously unseen examples



Full list at: ela.st/nlp-supported-models

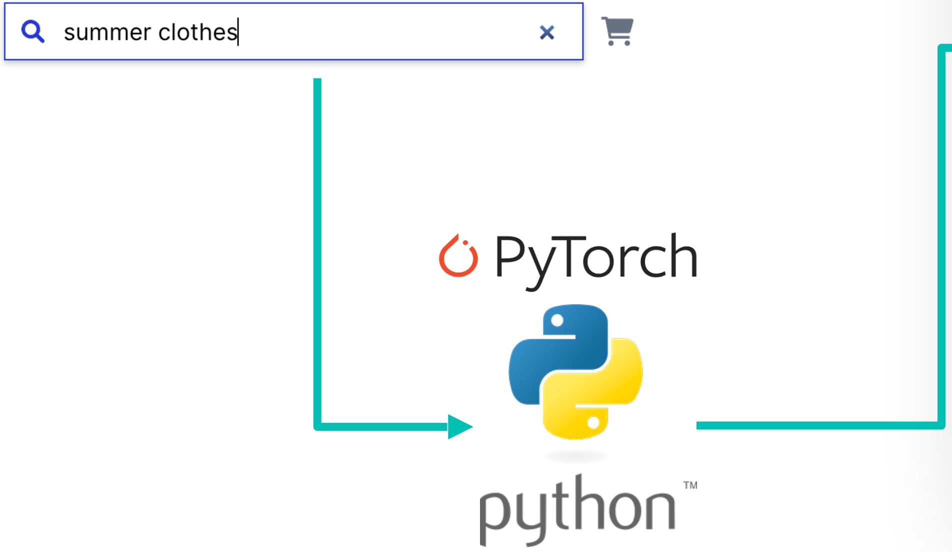
elastic

How do you

search **vectors**?

elastic

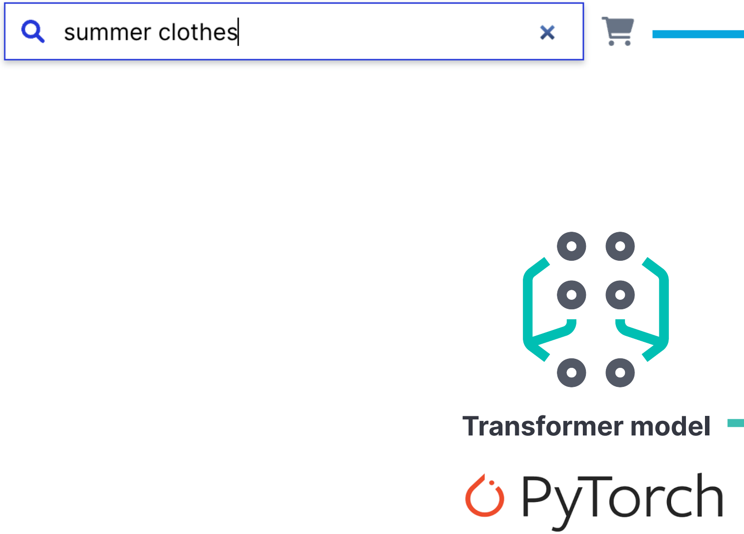
Vector Query




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

Vector Query

Commercial




```
GET product-catalog/_search
{
 "query" : {
    "bool": {
      "must": [{
         "knn": {
           "field": "desc_embbeding",
           "num_candidates": 50,
           "query_vector_builder": {
             "text_embedding": {
               "model_text": "summer clothes",
               "model_id": <text-embedding-model>
             }
           }
         }
      }],
      "filter": {
        "term": {
          "department": "women"
        }
      }
    }
  },
  "size": 10
}
```

elastic

Vector Search
components

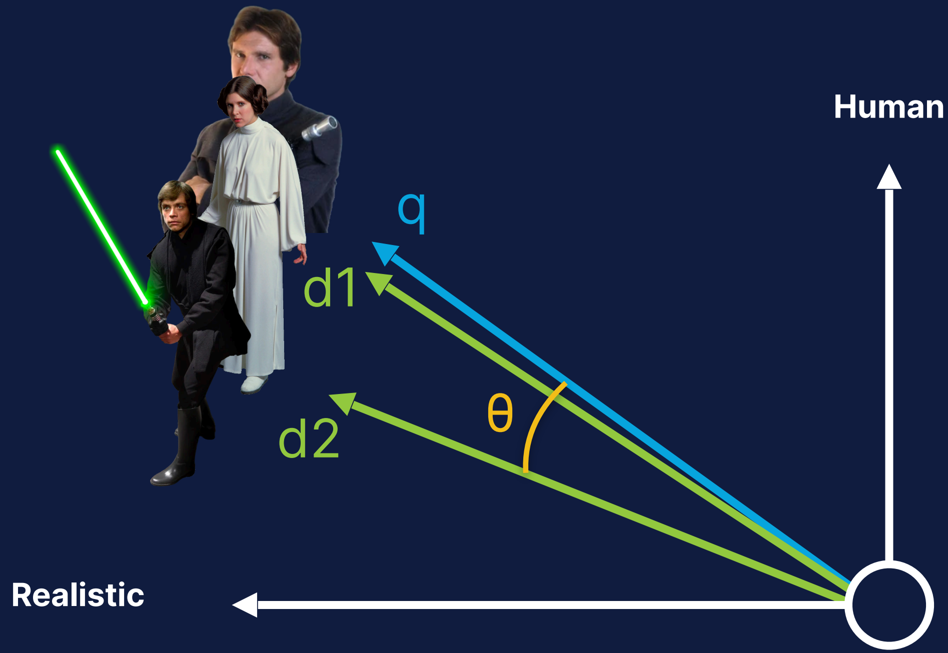
Search	Index	Generate
Query	**Mapping**	**Embedding**
`kNN`	`dense_vector`	Text embedding model (3rd party, local, in Elasticsearch)

elastic

But how does it

really work?

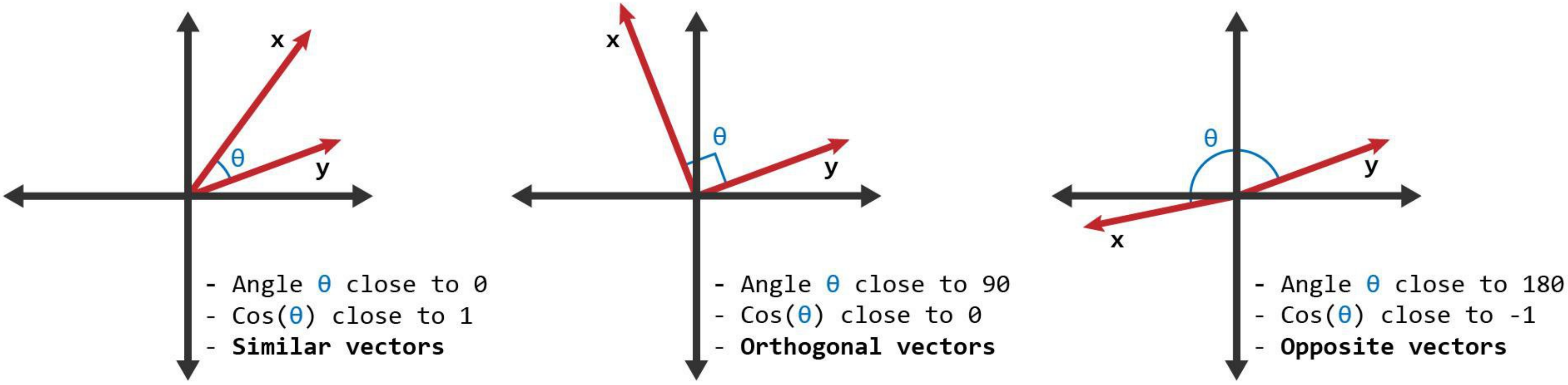
Similarity: cosine (cosine)



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

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

Similarity: cosine (cosine)



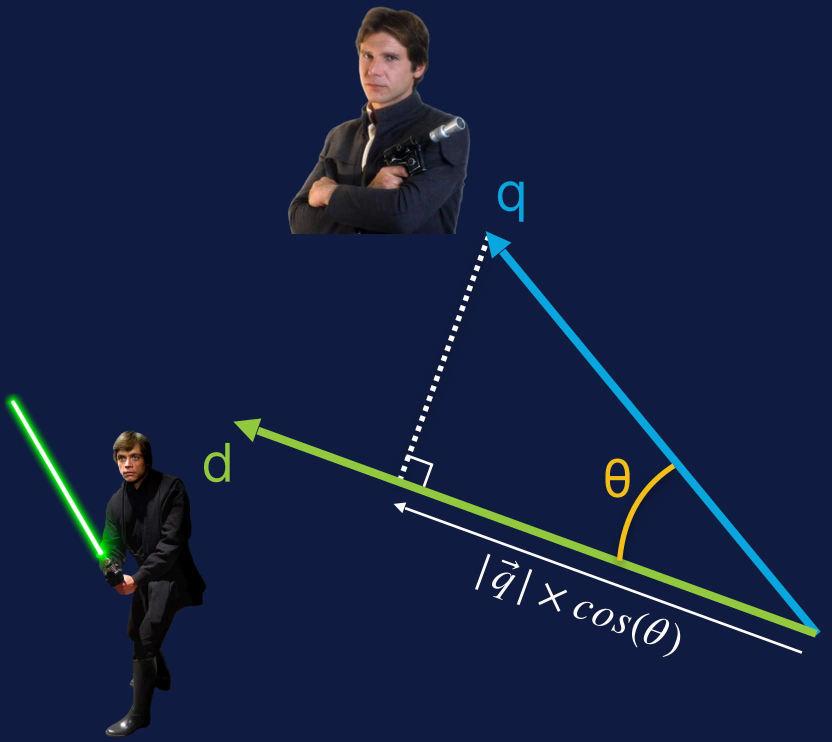
$$\_score = \frac{1+1}{2} = 1$$

$$\_score = \frac{1+0}{2} = 0.5$$

$$\_score = \frac{1-1}{2} = 0$$

elastic

Similarity: Dot Product (dot_product)

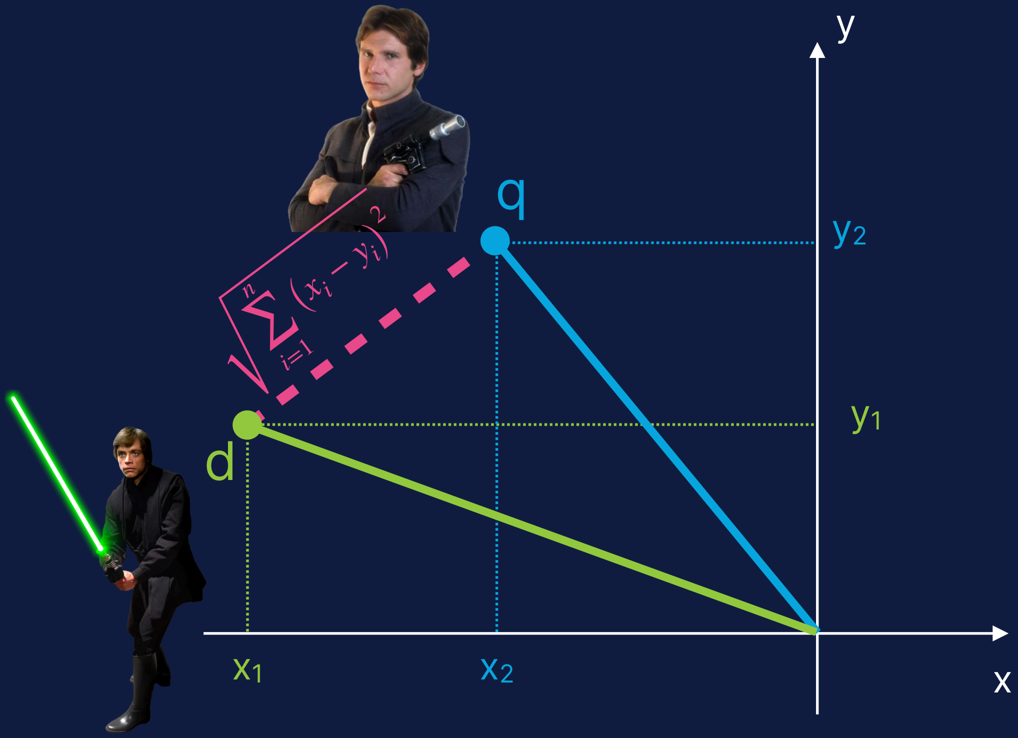


$$\vec{q} \times \vec{d} = |\vec{q}| \times cos(\theta) \times |\vec{d}|$$

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

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

Similarity: Euclidean distance (l2_norm)



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

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

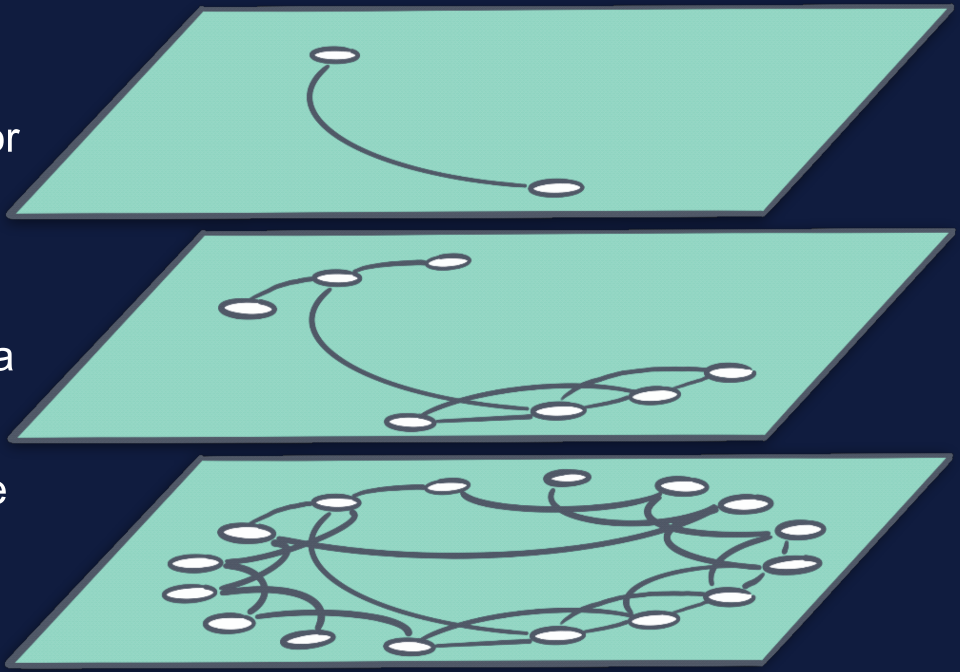
Brute Force

elastic

Hierarchical Navigable Small Worlds (HNSW)

One popular approach

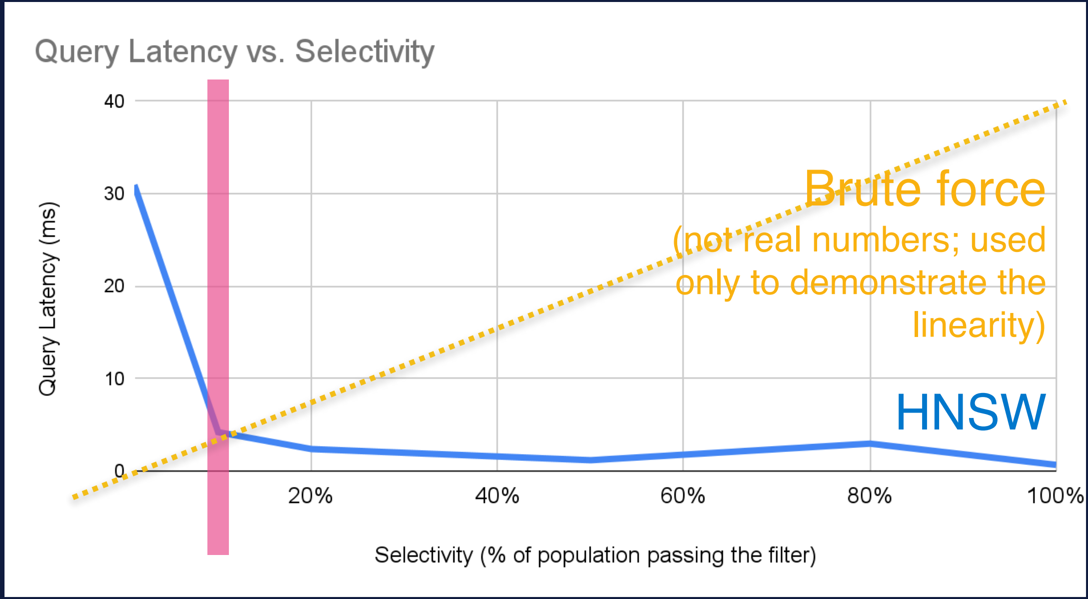
- **HNSW:** a layered approach that simplifies access to the nearest neighbor
- **Tiered:** from coarse to fine approximation over a few steps
- **Balance:** Bartering a little accuracy for a lot of scalability
- **Speed:** Excellent query latency on large scale indices



elastic

Filtering KNN Vector Similarity

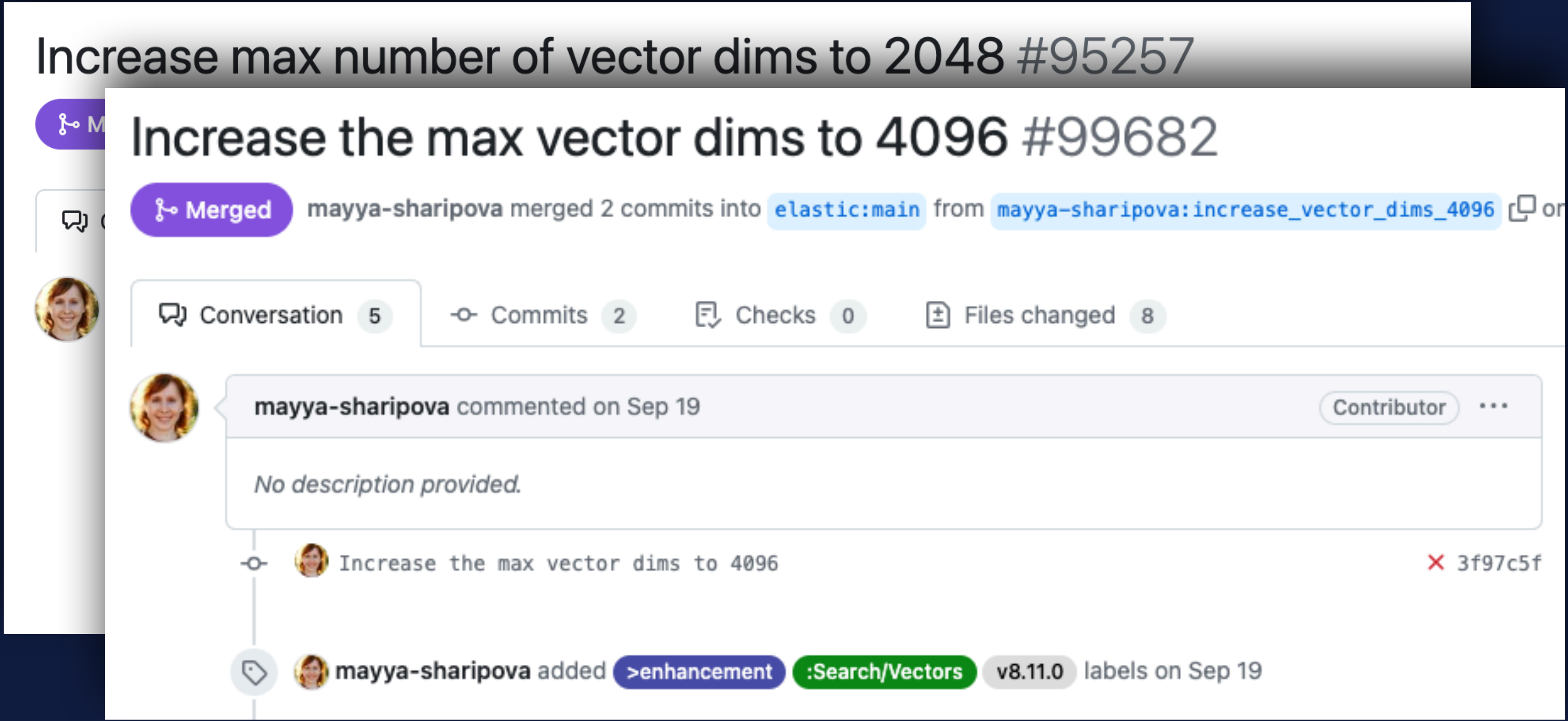
Automatically choose between brute force and HNSW



Bound worst case to 2*(brute force)

- Brute force scales O(n) of filtered
- HNSW scales ~O(log(n)) of all docs

Elasticsearch + Lucene = fast progress ❤️



elastic

Scaling Vector Search

Vector search

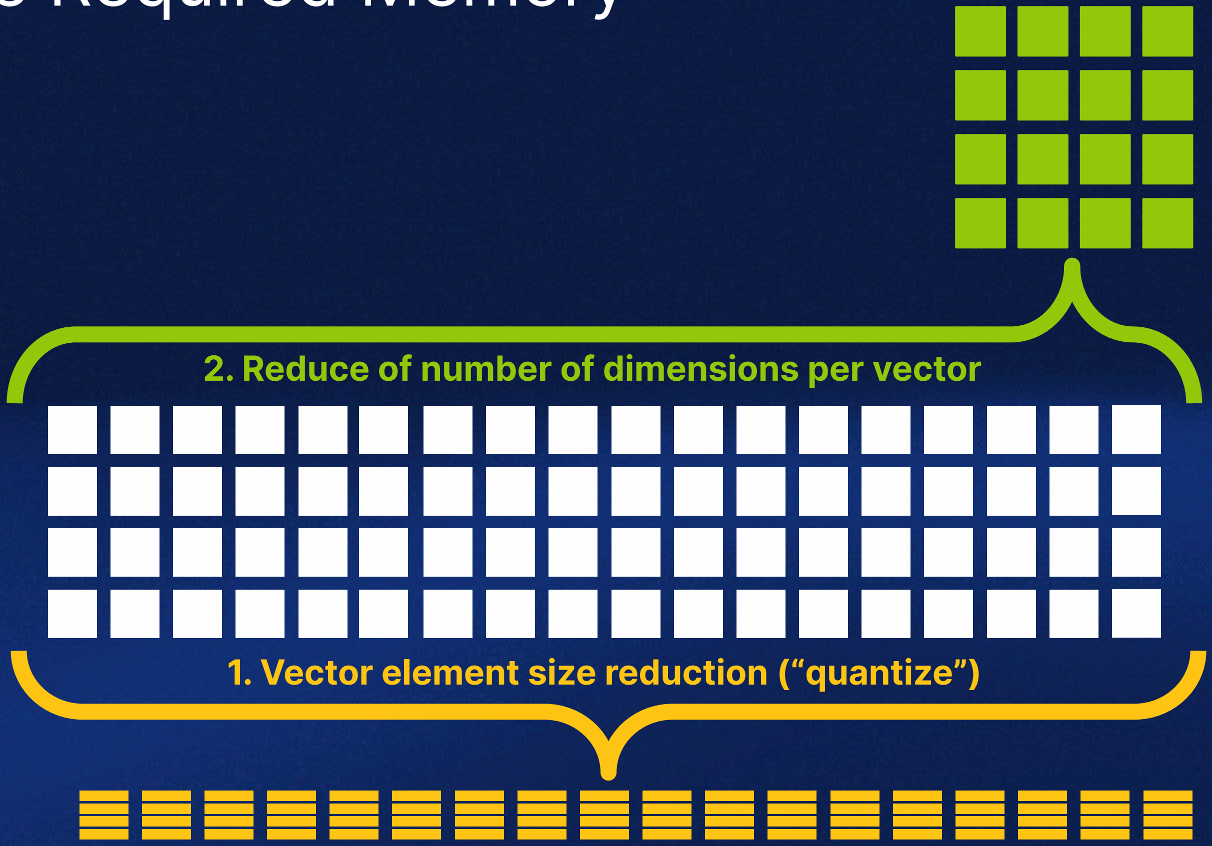
1. Needs lots of memory
2. Indexing is slower
3. Merging is slow

* Continuous improvements in Lucene + Elasticsearch

Best practices

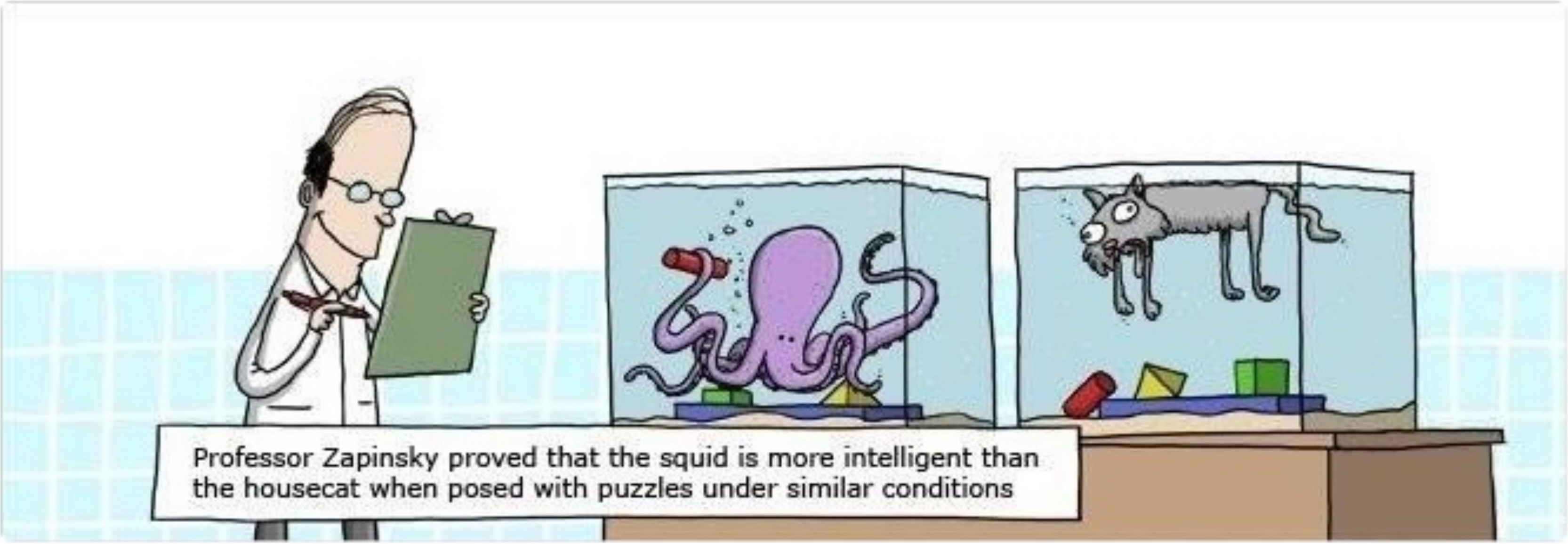
1. Avoid searches during indexing
2. Exclude vectors from _source
3. Reduce vector dimensionality
4. Use byte rather than float

Reduce Required Memory



elastic

Benchmarketing



elastic

https://github.com/erikbern/ann-benchmarks

Add Elasticsearch KNN #401

Merged **erikbern** merged 1 commit into `erikbern:main` from `ceh-forks:elasticsearch` last week

Conversation 5 | Commits 1 | Checks 34 | Files changed 4

ceh commented last week • edited — Contributor

Add an implementation for Elasticsearch KNN search. Fixes #298.

- Supports specifying `index_options` via arg groups, with one entry for Elasticsearch's default settings for M and EF.
- Supports specifying `num_candidates` via query args.

What do you think **@erikbern**, **@alexklibisz**?

👍 1

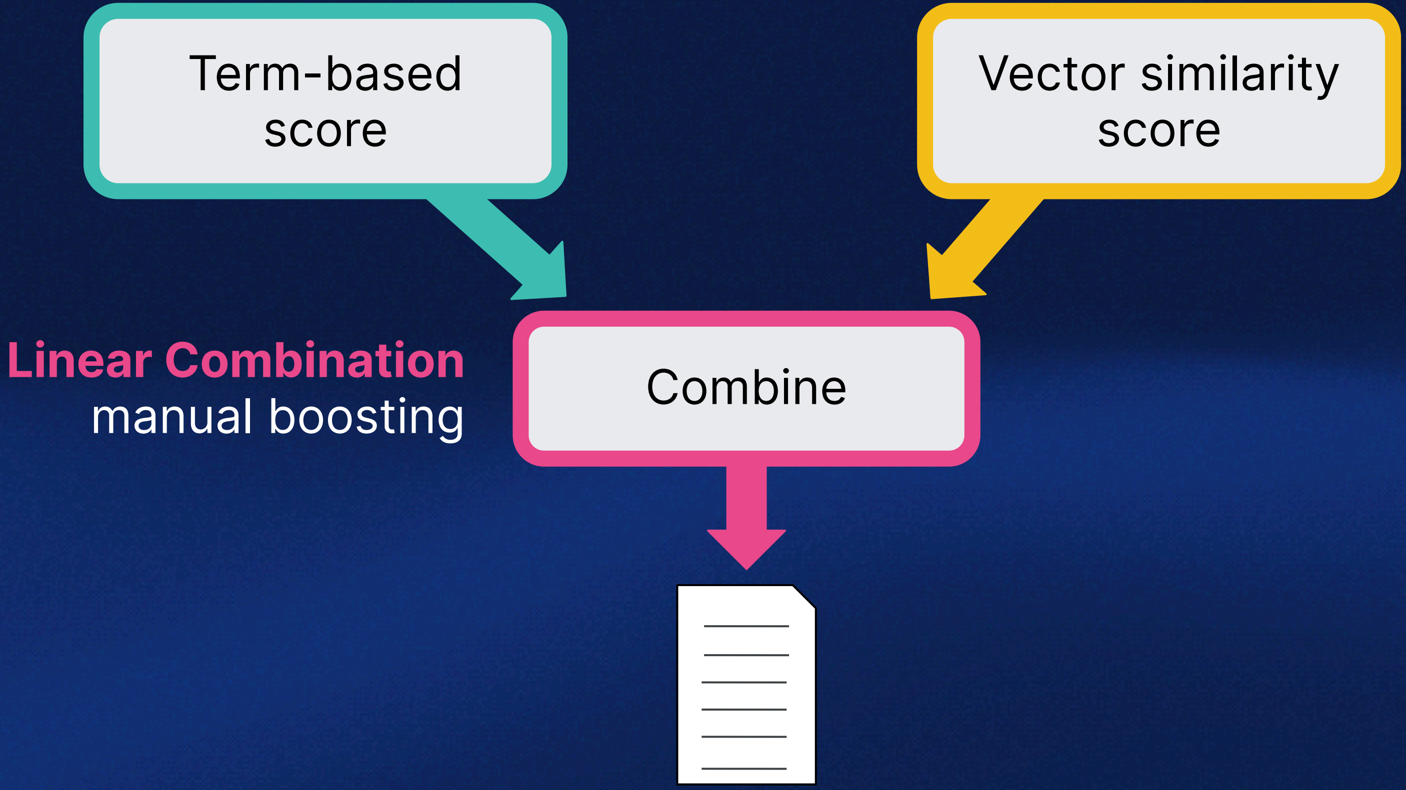
elastic

Elasticsearch

You Know, for **Hybrid** Search

elastic

Hybrid scoring



```
GET product-catalog/_search
{
  "query" : {
    "bool" : {
      "must" : [{
        "match": {
          "description": {
            "query": "summer clothes",
            "boost": 0.9
          }
        }
      },{
        "knn": {
          "field": "desc_embbeding",
          "query_vector": [0.123, 0.244,...],
          "num_candidates": 50,
          "boost": 0.1,
          "filter": {
            "term": {
              "department": "women"
            }
          }
        }
      }],
      "filter" : {
        "range" : { "price": { "lte": 30 } }
      }
    }
  }
}
```

summer clothes

pre-filter

post-filter

elastic

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

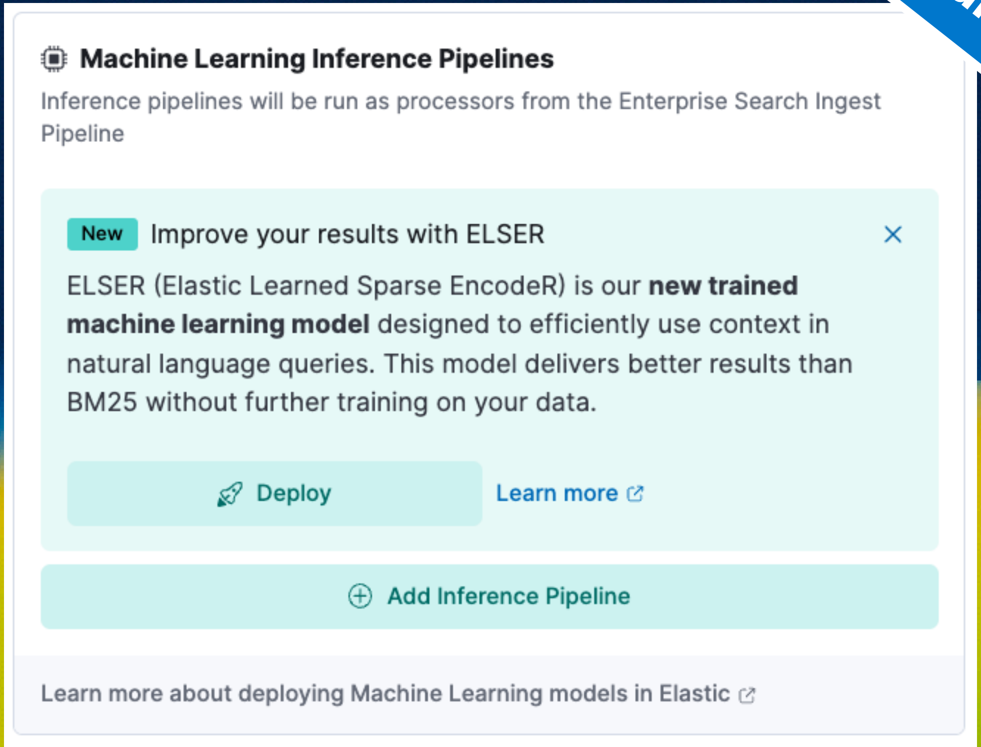
ELSER

Elastic **L**earned **S**parse Encod**ER**

Commercial

text_expansion

- Not BM25 or (dense) vector
- Sparse vector like BM25
- Stored as inverted index



Commercial

```
PUT /_inference/sparse_embedding/my_elser_model
{
  "service": "elser",
  "service_settings": {
    "num_allocations": 1,
    "num_threads": 1
  },
  "task_settings": {}
}
```

```
PUT /_inference/text_embedding/openai_embeddings
{
    "service": "openai",
    "service_settings": {
        "api_key": "<api_key>"
    },
    "task_settings": {
       "model": "text-embedding-ada-002"
    }
}
```

```
PUT /_inference/text_embedding/hugging_face_embeddings
{
    "service": "hugging_face",
    "service_settings": {
       "api_key": "<access_token>",
       "url": "<url_endpoint>"
    }
}
```

Commercial

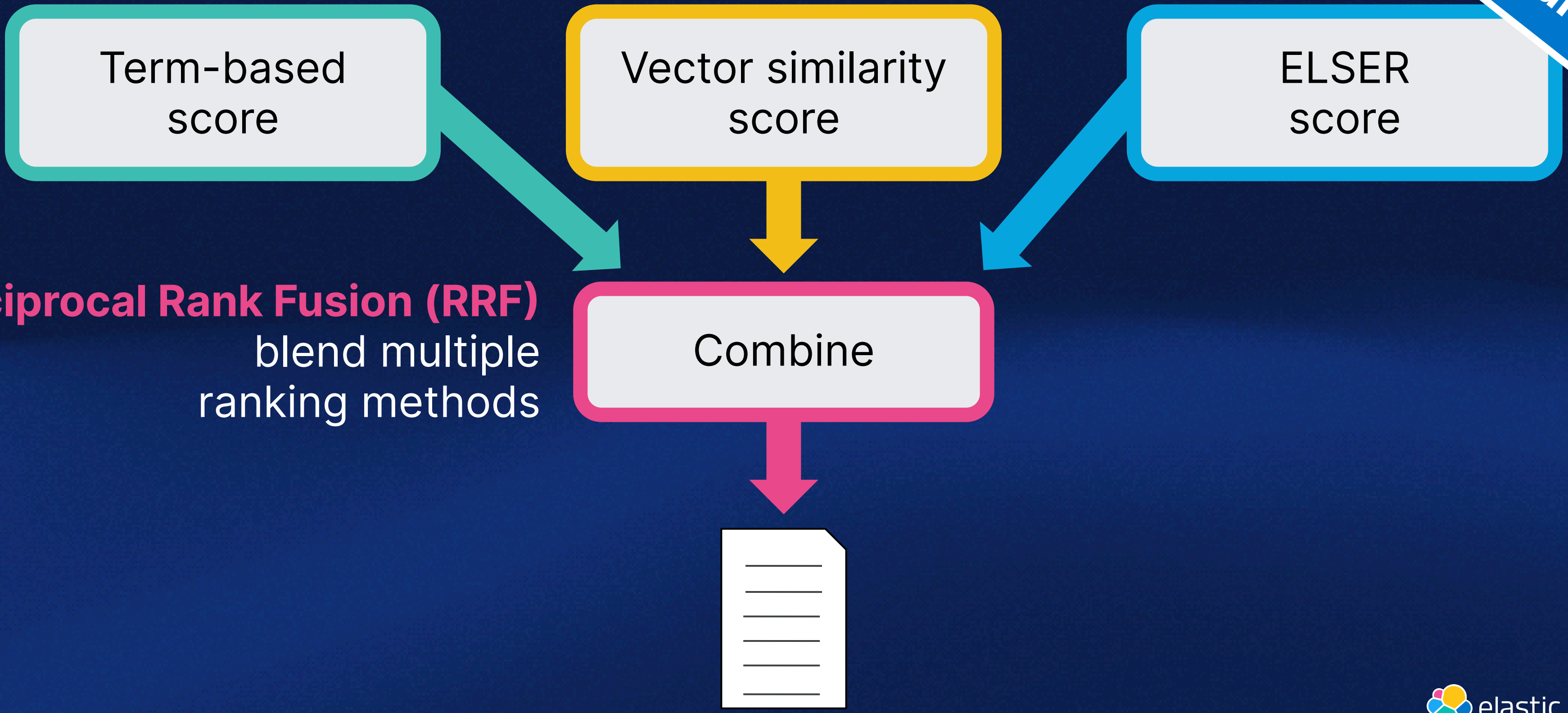
```
POST /_inference/sparse_embedding/my_elser_model
{
  "input": [
    "These are not the droids you are looking for.",
    "Obi-Wan never told you what happened to your father."
  ]
}
```

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

elastic

Hybrid ranking

Commercial



Reciprocal Rank Fusion (RRF)
blend multiple ranking methods

elastic

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

Commercial

BM25f

+

ELSER

+

Vector

Hybrid Ranking

elastic

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

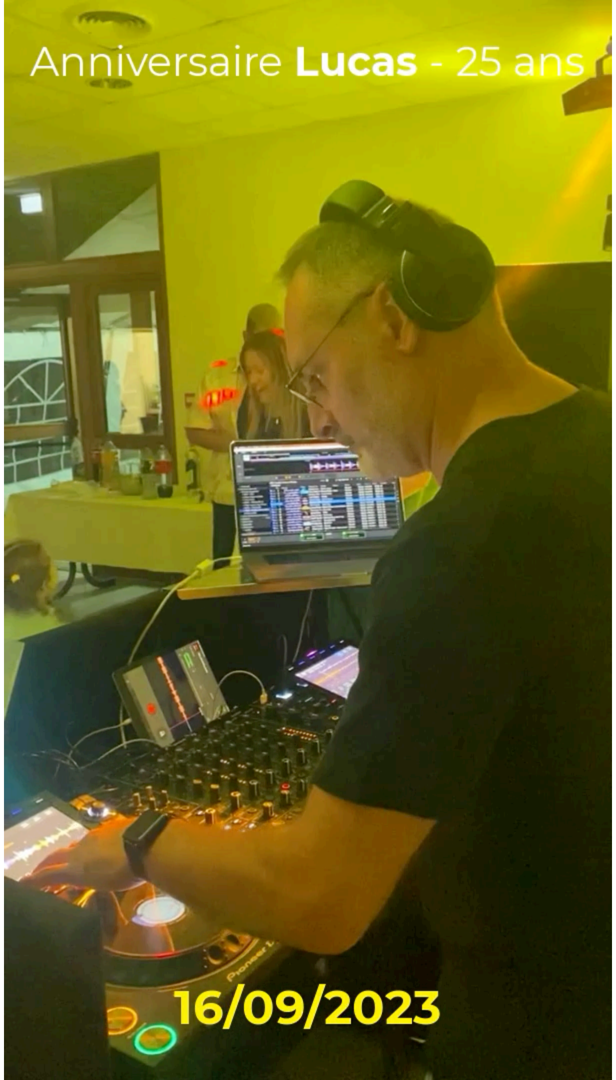
Ranking Algorithm 1			
Doc	Score	r(d)	k+r(d)
A	1	1	61
B	0.7	2	62
C	0.5	3	63
D	0.2	4	64
E	0.01	5	65

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

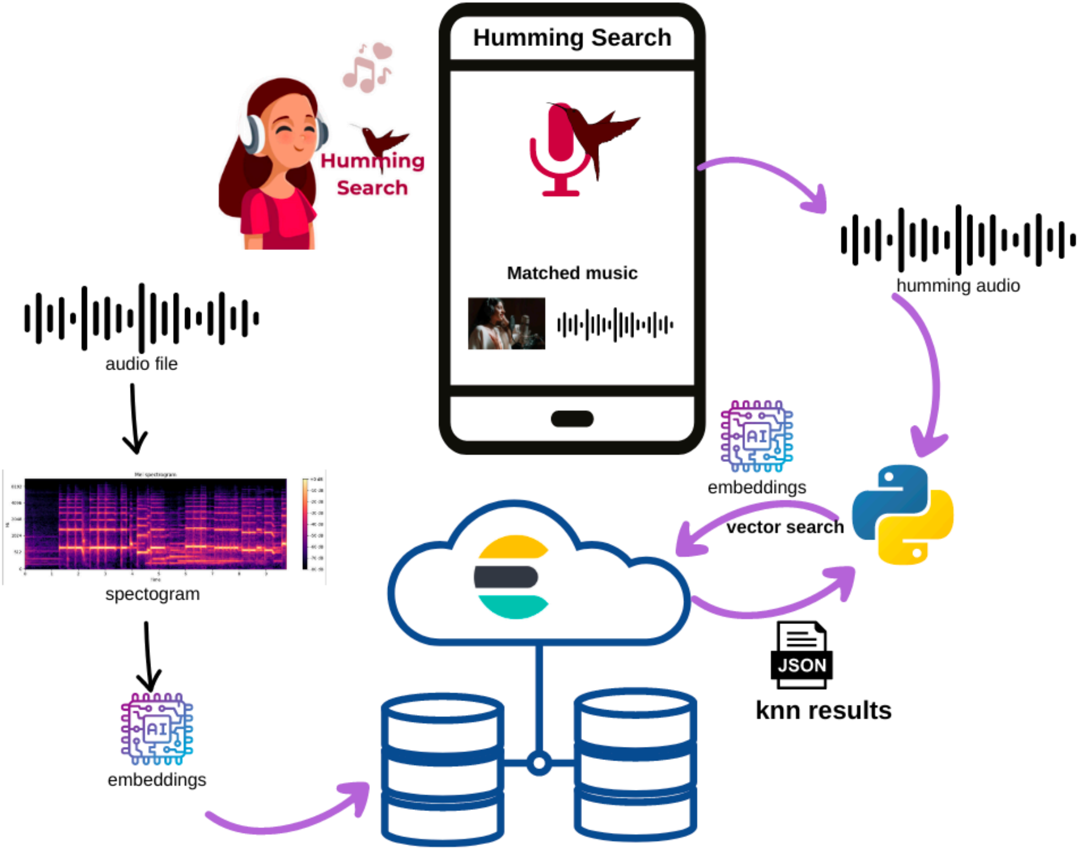
Doc	RRF Score
A	1/61 + 1/62 = 0,0325
C	1/63 + 1/61 = 0,0323
B	1/62 = 0,0161
F	1/63 = 0,0159
D	1/64 = 0,0156

elastic

https://djdadoo.pilato.fr/



elastic



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

elastic

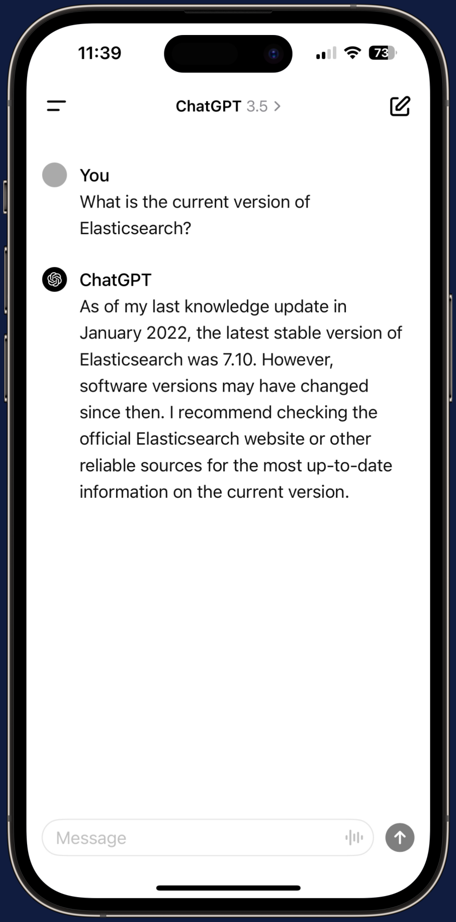
ChatGPT

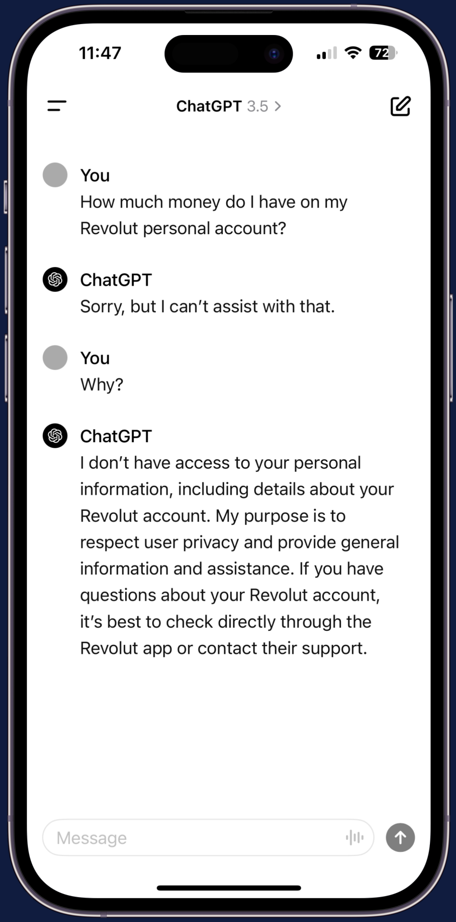
Elastic and LLM

elastic


Gen AI
Search engines

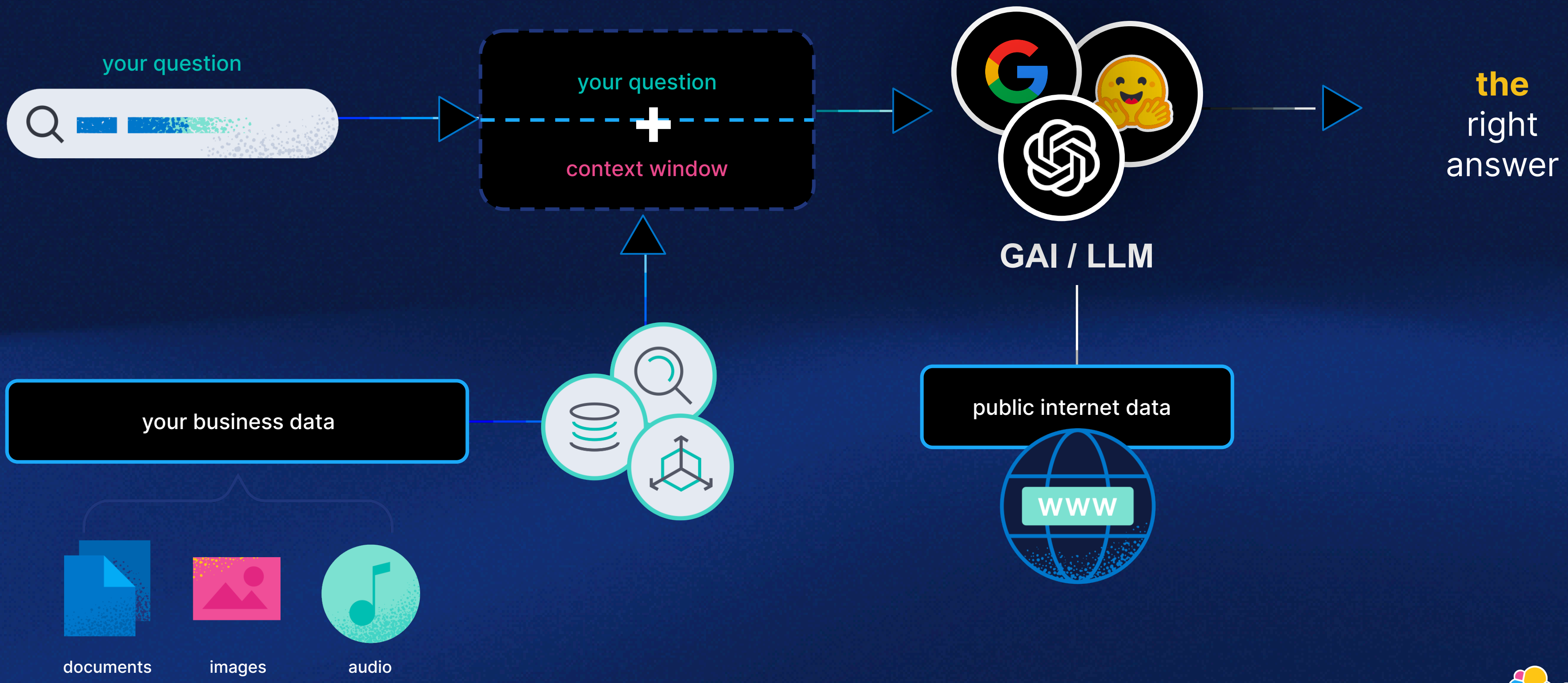
LLM: opportunities and limits


11:39
ChatGPT 3.5
You
What is the current version of Elasticsearch?
ChatGPT
As of my last knowledge update in January 2022, the latest stable version of Elasticsearch was 7.10. However, software versions may have changed since then. I recommend checking the official Elasticsearch website or other reliable sources for the most up-to-date information on the current version.
Message


11:47
ChatGPT 3.5
You
How much money do I have on my Revolut personal account?
ChatGPT
Sorry, but I can't assist with that.
You
Why?
ChatGPT
I don't have access to your personal information, including details about your Revolut account. My purpose is to respect user privacy and provide general information and assistance. If you have questions about your Revolut account, it's best to check directly through the Revolut app or contact their support.
Message

elastic

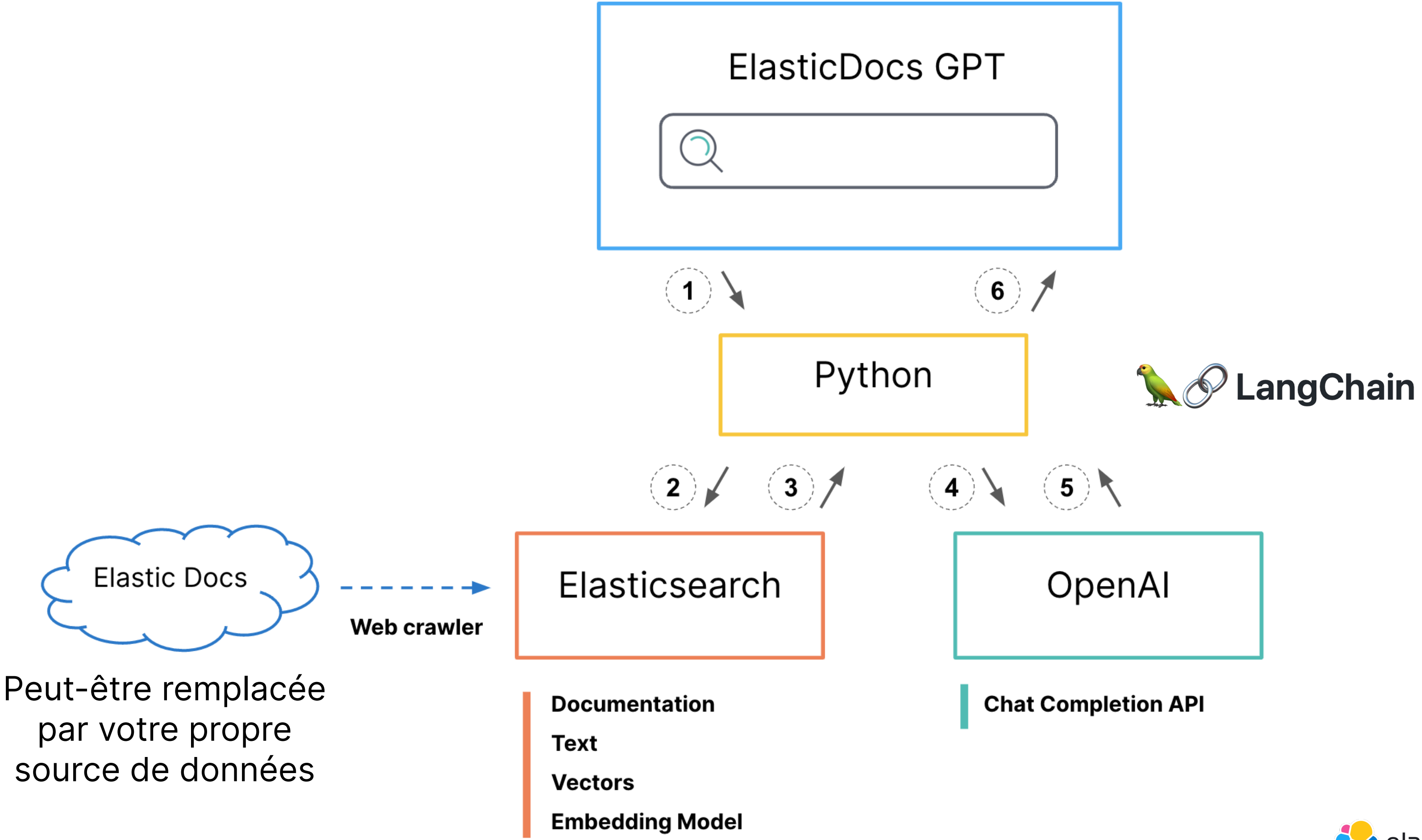
Retrieval Augmented Generation



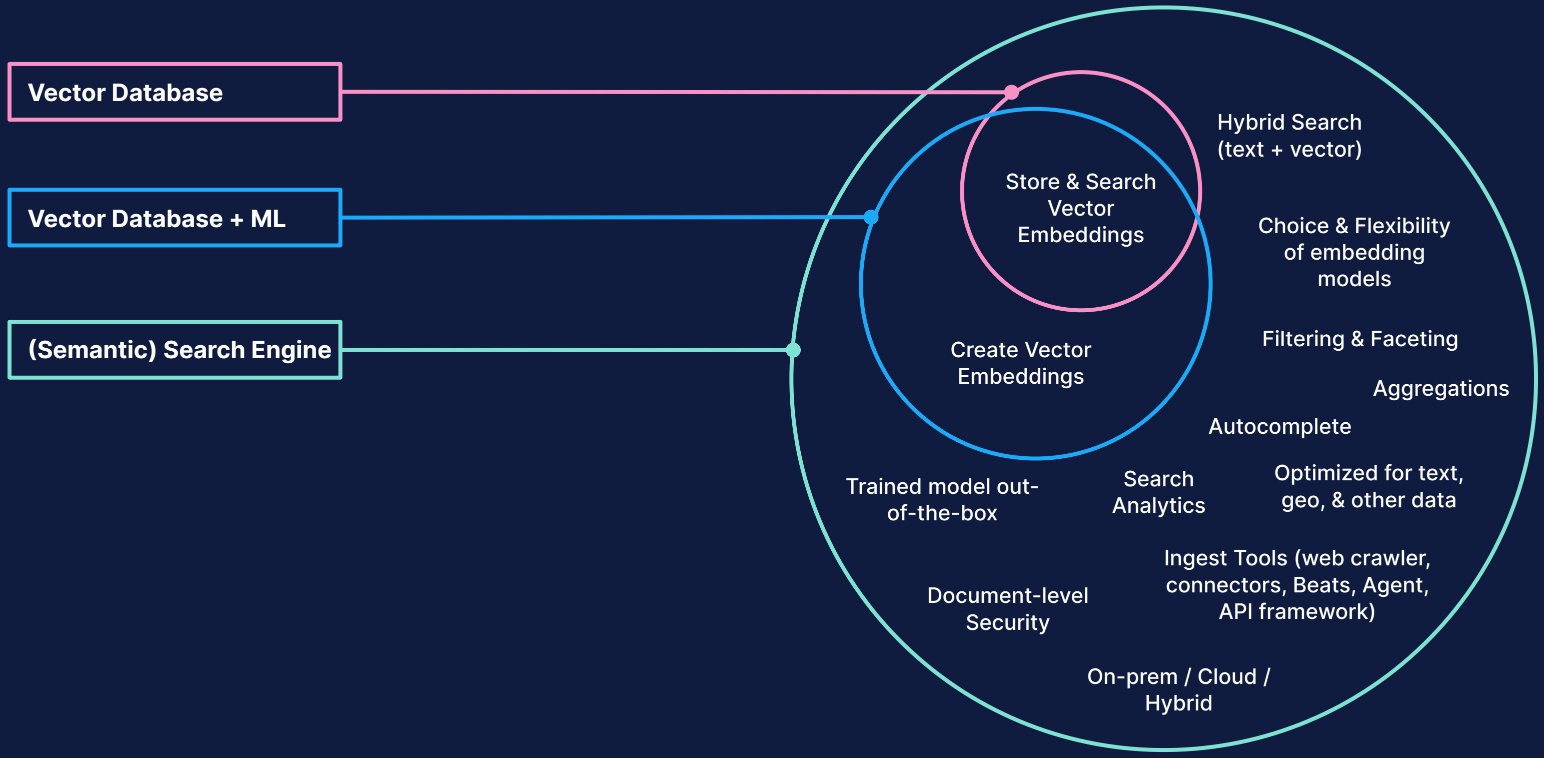
elastic

Demo

Elastic + Azure OpenAI
AWS Bedrock
Google Vertex AI


ElasticDocs GPT
1
6
Python
LangChain
2
3
4
5
Elastic Docs
Web crawler
Elasticsearch
OpenAI
Peut-être remplacée par votre propre source de données
Documentation
Text
Vectors
Embedding Model
Chat Completion API

Conclusion


Vector Database
Vector Database + ML
(Semantic) Search Engine
Store & Search Vector Embeddings
Hybrid Search (text + vector)
Choice & Flexibility of embedding models
Create Vector Embeddings
Filtering & Faceting
Aggregations
Autocomplete
Trained model out-of-the-box
Search Analytics
Optimized for text, geo, & other data
Document-level Security
Ingest Tools (web crawler, connectors, Beats, Agent, API framework)
On-prem / Cloud / Hybrid

Elasticsearch

You Know, for Semantic Search

elastic

elastic

Search & AI:
a new era

David Pilato | @dadoonet

LYON JUG