In partnership with FerretDB

Vector Search in Modern Databases

Peter Zaitsev,
Founder at Percona
July 1, 2024

Budapest, Hungary 2024

PERCONA UNIVERSITY

Open Source Databases Meetup

In partnership with FerretDB

Vector search support in databases

Opensource vector dbs

Milvus	2019
Vespa	2020
Weaviate	2021
Qdrant	2022

Opensource dbs and search engines

PostgreSQL	2021
Lucene	2021
Opensearch	2022
Redis	2022
SOLR	2022
Cassandra	**2023**
Typesense	**2023**
Clickhouse	**2023**
Manticore Search	**2023**
Meilisearch	**2023**
MariaDB	**In progress**
MySQL	Not yet

Non-opensource dbs

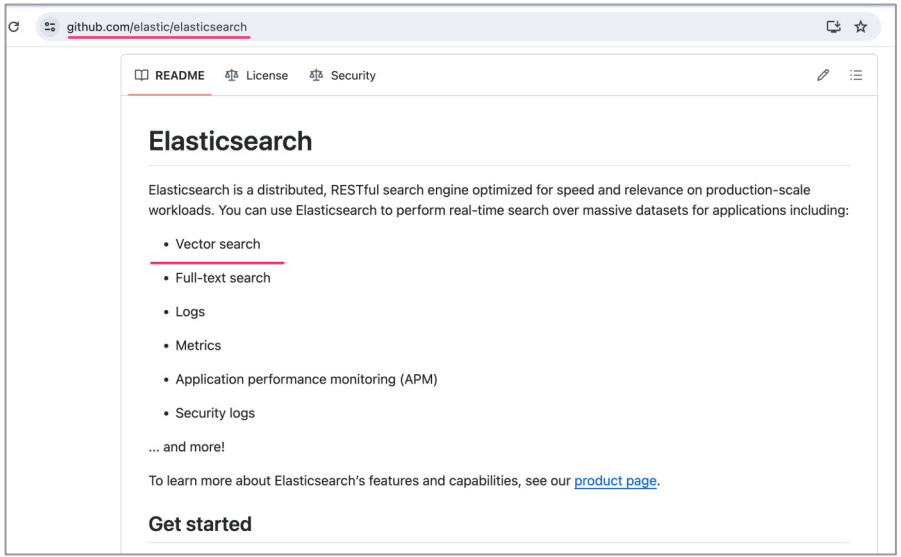
Elasticsearch	2019
Oracle	**2023**
MongoDB	**2023**

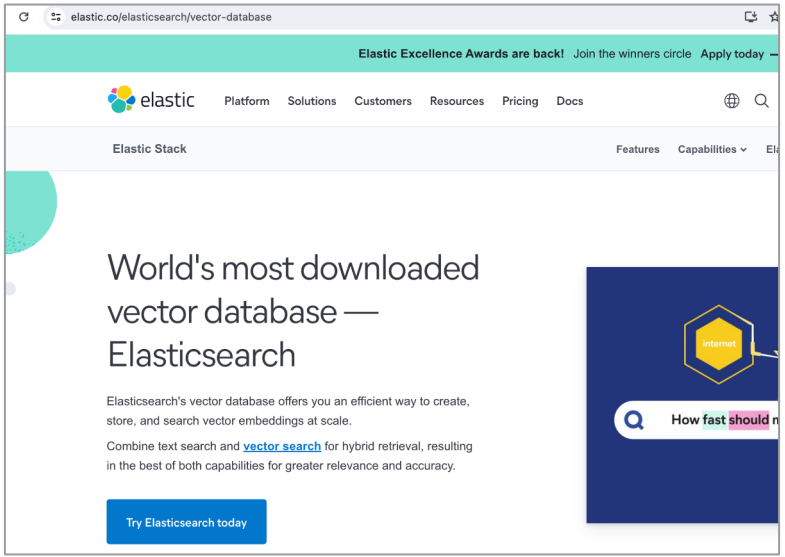
Clouds

Pinecone	2019
Amazon Elasticsearch / Opensearch	2020
Google Cloud Platform	2021
Alibaba Cloud AnalyticDB	**2023**
Azure	**2023**
Amazon DocumentDB	**2023**
Cloudflare Vectorize	**2023**

PERCONA

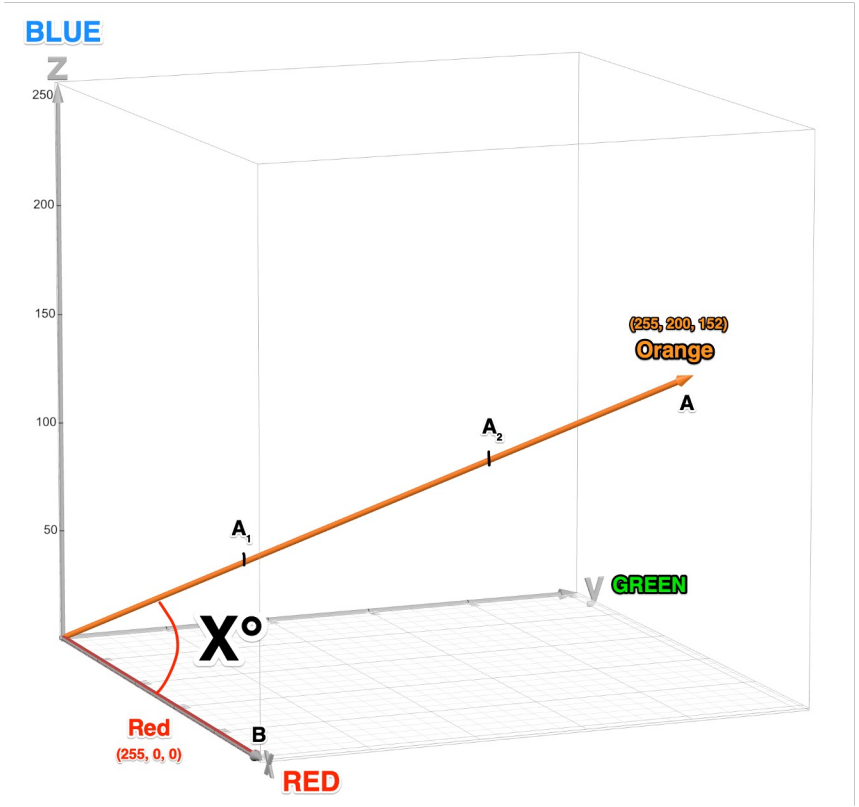
Vector search support in Elasticsearch





PERCONA

Vector space and vector similarity



- x : 0 .. 90°
- cos(x): 0 .. 1
- cos(x) is the same between B an A_1, A_2 and A
- Cosine similarity accounts vector lengths: 0 .. 1

$$\text{cosine similarity} = S_C(A, B) := \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|}$$

Vector space and vector similarity

From OpenAI API:

Which distance function should I use?

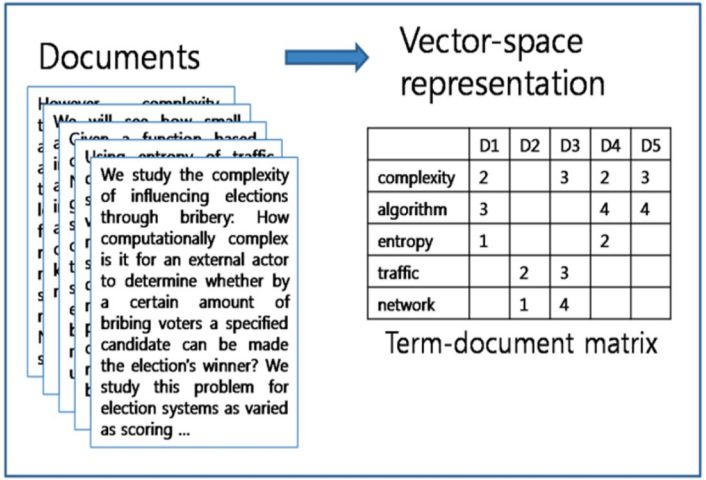
We recommend cosine similarity. The choice of distance function typically doesn't matter much.

OpenAI embeddings are normalized to length 1, which means that:

- Cosine similarity can be computed slightly faster using just a dot product
- Cosine similarity and Euclidean distance will result in the identical rankings

Vector features: sparse vectors

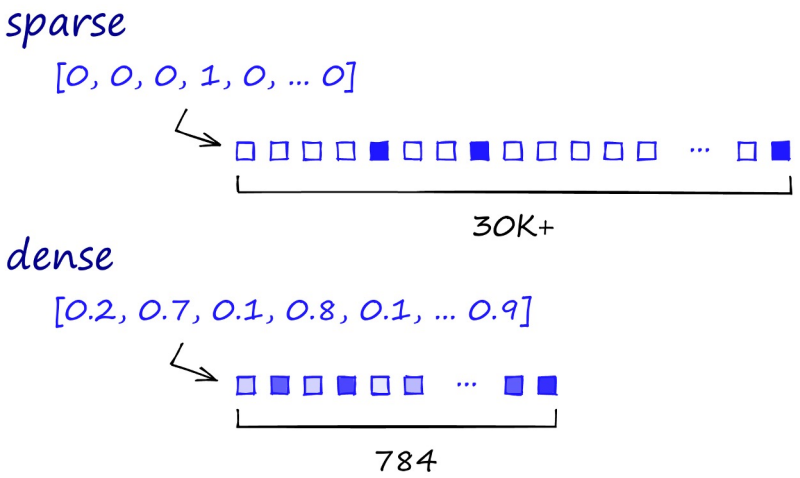
- [Green, Red, Blue]
- More dimensions?
- Bag of words sparse vectors:
  - [Has word “Hello”, has word “World”, …]
  - [Number of words “Hello”, number of words “World”, …]
  - [TF-IDF of word word “Hello”, TF-IDF of word word “World”, …]



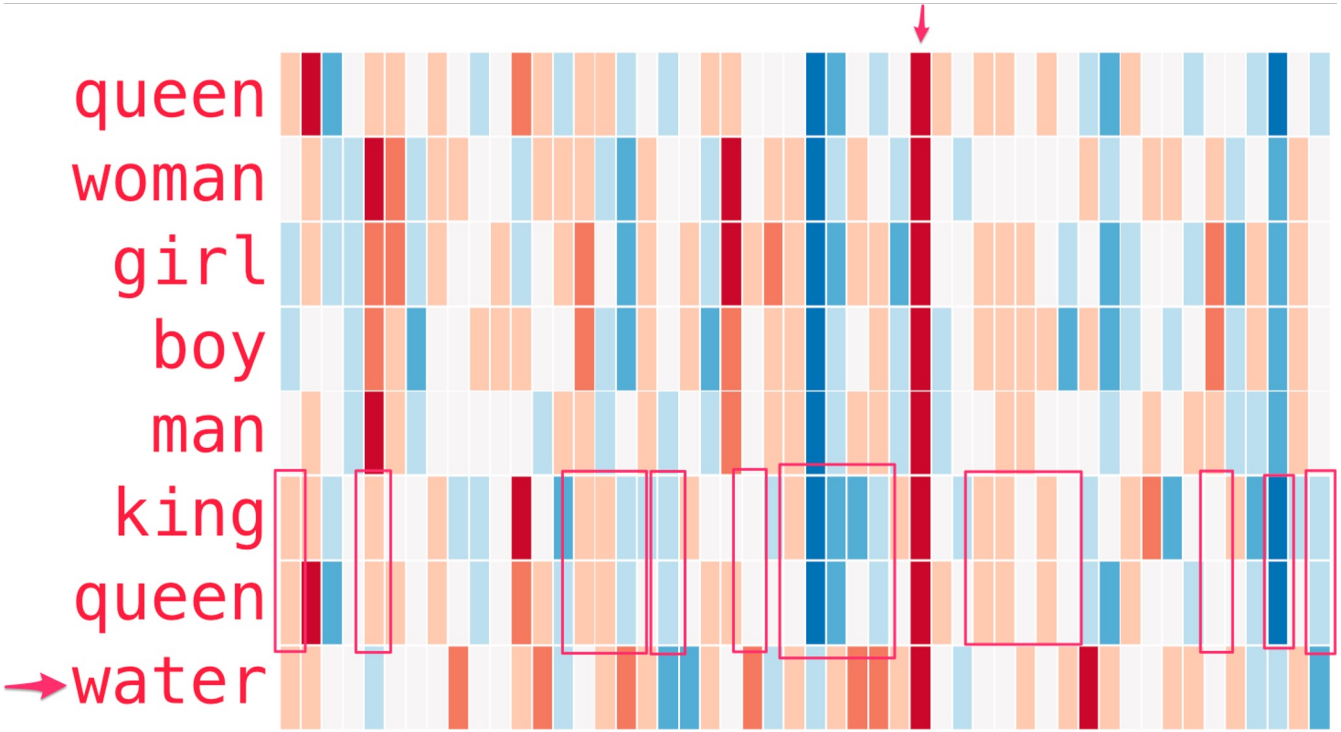
	D1	D2	D3	D4	D5
complexity	2		3	2	3
algorithm	3			4	4
entropy	1			2	
traffic		2	3		
network		1	4		

Vector features: dense vectors

- What's closer: a cat and a dog, or a cat and a car?
- Deep learning => embeddings:
  - Accounts contexts for texts: Word2vec, BERT, GPT
  - Vectors from images
  - Vectors from sounds



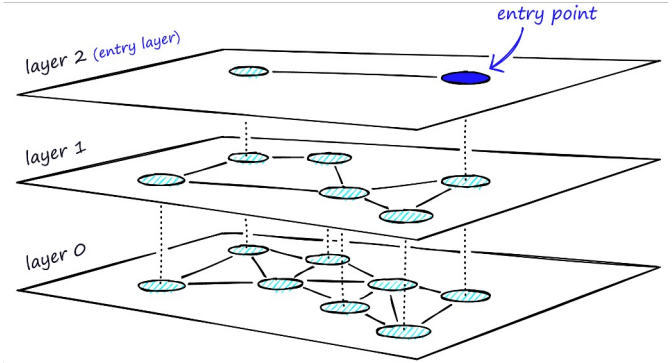
Embeddings



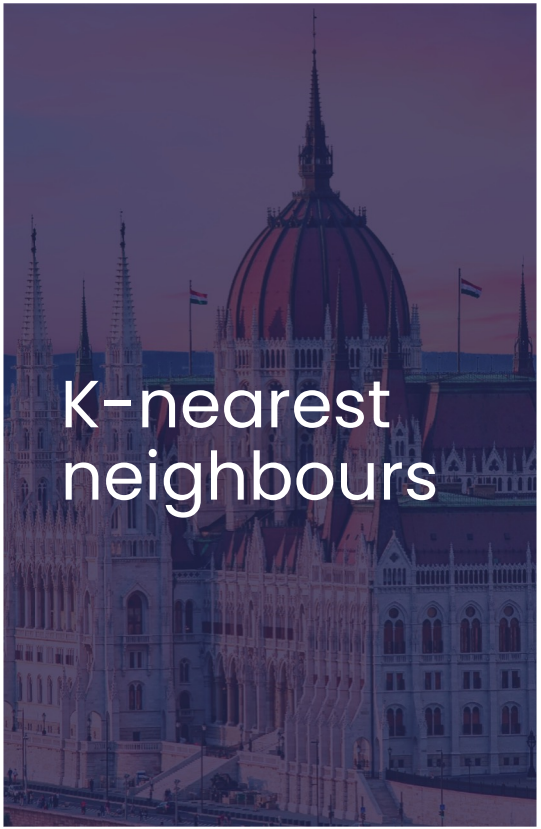
PERCONA

Dealing with embeddings

- Dense embeddings from deep learning pose indexing challenges.
- Traditional methods like inverted indexes are ineffective for non-sparse vectors.
- Dense vectors require comparison with all vectors in the dataset.
- Specialized indexes (KD-trees, LSH, HNSW, Annoy) enable:
  - Faster search
  - Insignificant accuracy loss.
- HNSW is used by most dbs and search engines: Postgres, Lucene, Opensearch, Redis, SOLR, Cassandra, Manticore Search, Opensearch and Elasticsearch, Typesense, Meilisearch

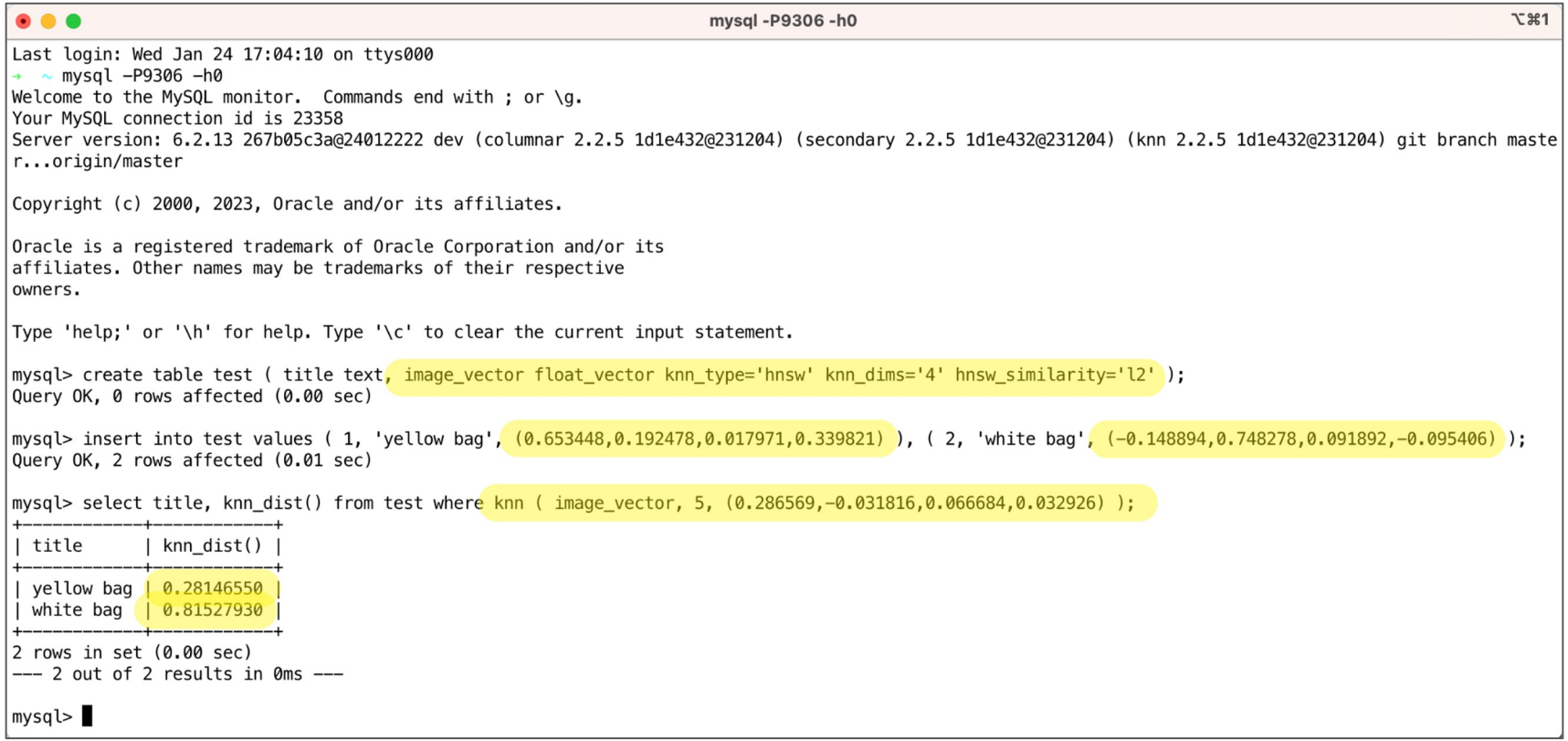


PERCONA



- Vector Search:
  - Clustering
  - Classification
  - KNN/ANN and more
- KNN and ANN - most attractive task in databases
  - Enhances databases with search engine-like features.

Vector search in dbs: typical implementation



Embedding computation

- Non-vector databases typically integrate external embeddings.
- Elasticsearch, Opensearch, Typesense enable automatic embedding generation.
- Microsoft's ONNX Runtime library can be used for integration (used by Vespa, Typesense)
- External embedding creation is challenging for users.
- So others are to catch up in embedding support.

Hybrid search approaches

- Typical solutions:
  - Reciprocal Rank Fusion

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

  - Multi-phase

BEIR Dataset	Vespa BM25	Vespa ColBERT	Vespa Hybrid
MS MARCO (*in-domain*)	0.228	**0.401**	0.344
TREC-COVID	0.690	0.658	**0.750**
NFCorpus	0.313	0.304	**0.350**
Natural Questions (NQ)	0.327	0.403	**0.404**
HotpotQA	0.623	0.298	**0.632**
FiQA-2018	0.244	0.252	**0.292**
ArguAna	0.393	0.286	**0.404**
Touché-2020 (V2)	0.413	0.315	**0.415**
Quora	0.761	0.817	**0.826**
DBPedia	0.327	0.281	**0.365**
SCIDOCS	0.160	0.107	**0.161**
FEVER	0.751	0.534	**0.779**
CLIMATE-FEVER	**0.207**	0.067	0.191
SciFact	0.673	0.403	**0.679**
Average nDCG@10 (excluding MS MARCO)	0.453	0.363	**0.481**

Conclusions

- Vector search is revolutionizing data retrieval, becoming common functionality of databases.
- Database Landscape Evolution:
  - Emergence of new vector-focused databases.
  - Established databases integrating vector search capabilities.
  - Reflects a growing demand for advanced search functions.
- Indexes like HNSW enhance speed.
- Future of Databases:
  - Transition from just supporting to internally generating embeddings.
  - Simplifying operations, enhancing power and intelligence.
  - Evolving from basic storage to systems that understand and analyze data.
- Paradigm Shift:
  - Vector search is a significant advancement in data management and retrieval.
  - Marks a new, exciting phase in the field.

In partnership with FerretDB