Vector search in modern databases

Peter Zaitsev, entrepreneur and co-founder of Percona

Sergey Nikolaev, CEO, Manticore
## Vector search support in databases

<table>
<thead>
<tr>
<th>Opensource vector dbs</th>
<th>Opensource dbs and search engines</th>
<th>Non-opensource dbs</th>
<th>Clouds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Milvus</td>
<td>PostgreSQL 2021</td>
<td>Elasticsearch 2019</td>
<td>Pinecone 2019</td>
</tr>
<tr>
<td>Vespa</td>
<td>Lucene 2021</td>
<td>Oracle 2023</td>
<td>Amazon Elasticsearch / Opensearch 2020</td>
</tr>
<tr>
<td>Weaviate</td>
<td>Opensearch 2022</td>
<td>MongoDB 2023</td>
<td>Google Cloud Platform 2021</td>
</tr>
<tr>
<td>Qdrant</td>
<td>Redis 2022</td>
<td></td>
<td>Alibaba Cloud AnalyticDB 2023</td>
</tr>
<tr>
<td></td>
<td>SOLR 2022</td>
<td></td>
<td>Azure 2023</td>
</tr>
<tr>
<td></td>
<td>Cassandra 2023</td>
<td></td>
<td>Amazon DocumentDB 2023</td>
</tr>
<tr>
<td></td>
<td>Typesense 2023</td>
<td></td>
<td>Cloudflare Vectorize 2023</td>
</tr>
<tr>
<td></td>
<td>Clickhouse 2023</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Manticore Search 2023</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Meilisearch 2023</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MariaDB In progress</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MySQL Not yet</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Clouds:
- Pinecone 2019
- Amazon Elasticsearch / Opensearch 2020
- Google Cloud Platform 2021
- Alibaba Cloud AnalyticDB 2023
- Azure 2023
- Amazon DocumentDB 2023
- Cloudflare Vectorize 2023
Vector search support in Elasticsearch

Elasticsearch

Elasticsearch is a distributed, RESTful search engine optimized for speed and relevance on production-scale workloads. You can use Elasticsearch to perform real-time search over massive datasets for applications including:

- Vector search
- Full-text search
- Logs
- Metrics
- Application performance monitoring (APM)
- Security logs

... and more!

To learn more about Elasticsearch's features and capabilities, see our product page.

Get started

World's most downloaded vector database — Elasticsearch

Elasticsearch’s vector database offers you an efficient way to create, store, and search vector embeddings at scale. Combine text search and vector search for hybrid retrieval, resulting in the best of both capabilities for greater relevance and accuracy.

Try Elasticsearch today
Vector space and vector similarity

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

$$\text{cosine similarity} = S_C(A, B) := \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$
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 “Hello”, TF-IDF of word word “World”, … ]
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

**sparse**

\[ [0, 0, 0, 1, 0, ... 0] \]

**dense**

\[ [0.2, 0.7, 0.1, 0.8, 0.1, ... 0.9] \]
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
K-nearest neighbours

- 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
    \[ RRF_{score}(d \in D) = \sum_{r \in R} \frac{1}{k + r(d)} \]
  - Multi-phase
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.