Engineering Lab 7 - Vector Database

Use a combination of tree-based and hashing-based methods to index high-dimensional vectors, facilitating quick lookups and similarity searches.

Problem Statement

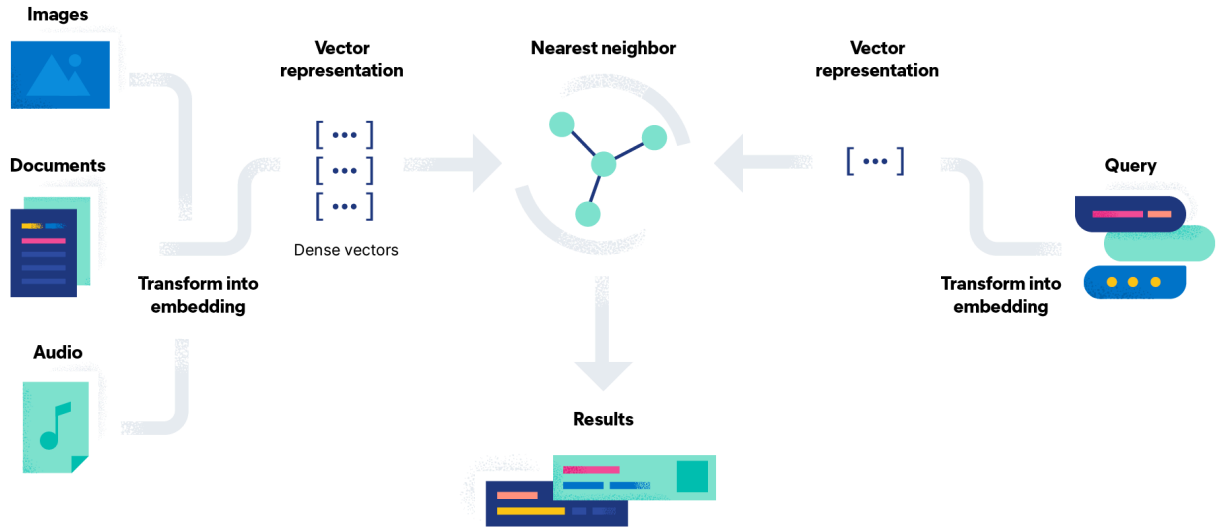
With the surge of multimedia content on the internet, it has become increasingly challenging to quickly search and retrieve relevant content based on similarity. Traditional relational databases aren't efficient for tasks such as finding the nearest neighbors in a high-dimensional space, which is crucial for applications like image search, recommendation systems, etc.

Objective

To design and implement a vector database that allows efficient storage, search, and retrieval of high-dimensional vectors, thereby enabling similarity searches and aiding applications that require finding items in close proximity in the vector space.

Solution

Overall Architecture and Constraints



Source: https://www.elastic.co/what-is/vector-database

1. **Database Structure:** A hybrid combination of KD-Trees for partitioning the space and Locality Sensitive Hashing (LSH) for bucketing similar items.

2. Scalability: The database should be scalable, able to handle large datasets, and support distributed architectures. Use sharding techniques to distribute data across multiple servers.

3. Robustness: The system should be resilient to failures. Implement a replication mechanism to create copies of data.

4. **Query Speed:** Aims for sub-second query speeds for nearest-neighbor lookups.

5. **Input Constraints:** Vectors should have a fixed length and be normalized.

References

- What is a Vector Database?
- C++ QuickStart tutorial
- Beginner Guide with Python
- Vector DB From Scratch Video

Stack and Technologies Used

1. Containerization
    a. Docker
    b. Kubernetes
2. Programming Languages
    a. Python (Entry)
    b. Go
    c. Rust (try?)
    d. C++ (try?)
3. ML Frameworks
    a. SciPy
    b. Pytorch
4. Database Management
    a. FoundationDB
    b. FAISS
        i. FAISS and sentence-transformers in 5 Minutes
    c. Milvus

Outcome

The vector database successfully indexes high-dimensional vectors and retrieves relevant vectors with sub-second latency. It showcases a marked improvement over traditional

methods in applications like image and audio similarity search, providing a more relevant and efficient search experience. This database can be used as a backbone in a myriad of applications, from recommendation engines to anomaly detection systems.