Introduction to Vector DataBases

Referred Lecture : Starting with vector databases

1. Introduction to Vector Databases

- Definition: A vector database stores data in the form of embeddings, which are sets of coordinates in a multidimensional space representing semantic information.

- Purpose: To enable semantic searching by allowing the retrieval of similar data based on embeddings.

2. Data Processing Steps

- Raw Data Extraction: Data typically resides in a data store.

- Semantic Data Extraction: Convert raw text into a semantic representation.

- Embedding Creation:

  - Tokenization: Break down text into tokens.

  - Embedding Model: Processes tokens and outputs a vector (embedding).

3. Internal Structure of a Vector Database

- Multidimensional Space: Embeddings are stored as coordinates.

  - Example: Clusters for fruits and vegetables categorized by similarity.

- Searching: Input a query embedding to find the nearest matches in the vector database.

4. Challenges with Embeddings

- Size Limitations: Embeddings have limits on input token size and output capacity.

- Chunking Data: To maintain context, large documents need to be broken into smaller, meaningful sections.

Chunking Strategies

1. Basic Chunking Approaches

- Fixed-Length Sections: Simple method but often loses context.

- Document Structure-Based: Works for stratified documents (e.g., how-to manuals).

2. Advanced Chunking Techniques

- Overlapping Chunks: Maintain continuity by overlapping sections to preserve context.
- Summarization:
  - Use document abstracts or generative models to summarize chunks.
  - Include summaries in embeddings to maintain the document's overall context.

Practical Implementation

1. Indexing Strategies

- Cosine Similarity: Common indexing method for vector data.

2. Tools for Vector Searches

- Elasticsearch: Integrates vector capabilities with traditional search.
- Specialized Vector Databases:
  - FAISS: Facebook AI Similarity Search.
  - Milvus: Fast, simple to use, and supports large clusters.
  - Weaviate: Feature-rich, supports GraphQL and RESTful APIs.
  - Pinecone: Hosted vector database service.
- Libraries:
  - Annoy: Python library for in-memory vector searches.
- Traditional RDBMS:
  - Postgres with PgVector: Adds vector search capabilities.
  - SQL Server 2022: Incorporates vector search functions.

Example Use Case

- Document: The Constitution of the United States.
  - Chunking: Based on natural document structure (e.g., articles, sections).
  - Embedding Model: BERT used for generating embeddings.

Conclusion

- Key Takeaways:
  - Understanding vector databases involves data chunking strategies and embedding generation.
  - Choosing the right tools and indexing methods is essential for effective semantic search.
- Further Exploration on the field of RAG (Retrieval-Augmented Generation) applications and their data pipelines

their data pipelines.