June 2024

Retrieval Augmented Generation with Atlas Vector Search (Simple)

Yu Chang Ou

LLM Basics

Q&A Style

What is an LLM?[7]

A Large Language Model (LLM) is an advanced **neural net** that uses the **Transformer** architecture to understand and generate human-like text by predicting the next word in a sentence based on vast amounts of text data it has been trained on.

What is a Neural Net?[8]

A **neural network** is a machine learning program, or model, that makes decisions in a manner similar to the human brain, by using processes that mimic the way biological neurons work together to identify phenomena, weigh options and arrive at conclusions.

How does a Neural Net work?

A neural net works by passing data through layers of neurons, where each connection has **weights** and **biases** that adjust to minimize errors and improve accuracy in tasks like prediction and classification.

How does this relate to LLM?

Fundamentally, LLMs such as ChatGPT and BERT uses **Transformer Neural Networks**. GPT even stands for “Generative Pre-trained Transformer”.

What is a vector embedding?

Embeddings are continuous vector representations of words or tokens that encode their semantic meanings in a high-dimensional space.

These numerical representations convert diverse data types, including words and sentences, allowing efficient data processing and comprehension by machines.

What is Vector Search?[13]

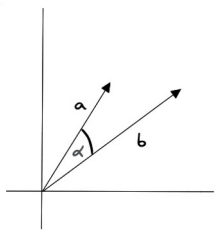
Vector search is a search method that returns results based on your data's semantic, or underlying, meaning.

Unlike traditional full-text search which finds text matches, vector search finds vectors that are close to your search query in multi-dimensional space. The closer the vectors are to your query, the more similar they are in meaning.

What are vector embeddings?

Vector embeddings, or vectorization, is the process of converting your data into vectors. These embeddings capture meaningful relationships in your data and enable tasks like semantic search and retrieval. To use Atlas as a vector database, you create embeddings by passing your data through an embedding model, and you store these embeddings as a field in the document.

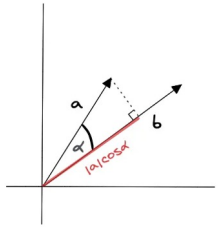
How to Measure Vector Similarity?[14]



Cosine

measures similarity based on the angle between vectors.

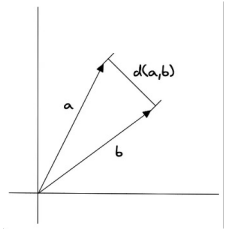
Use cosine similarity is solving semantic search and document classification problems since it allows you to compare the direction of the vectors. Similarly, recommendation systems that aim to recommend items to users based on their past behavior could use this similarity metric.



Dot Product

measures similar to cosine, but takes into account the magnitude of the vector.

Dot Product is good for recommender system. if two products have embeddings with the same direction but different magnitudes, this can mean that the two products are about the same topic, but the one that has a larger magnitude is just better / more popular than the other.

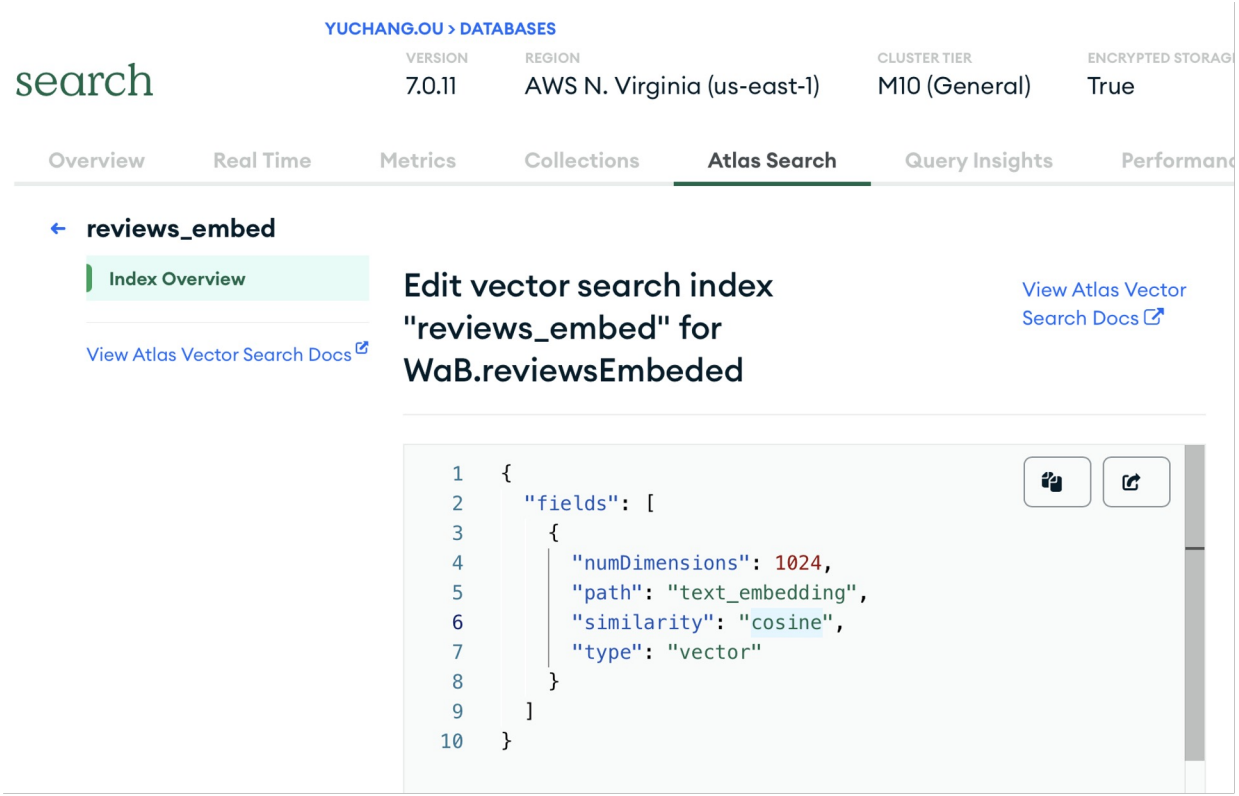


Euclidean

measures the distance between ends of vectors.

Since Euclidean distance is sensitive to magnitudes, it is helpful where the embeddings contain information that has to do with counts or measures of things.

Atlas Vector Search Index



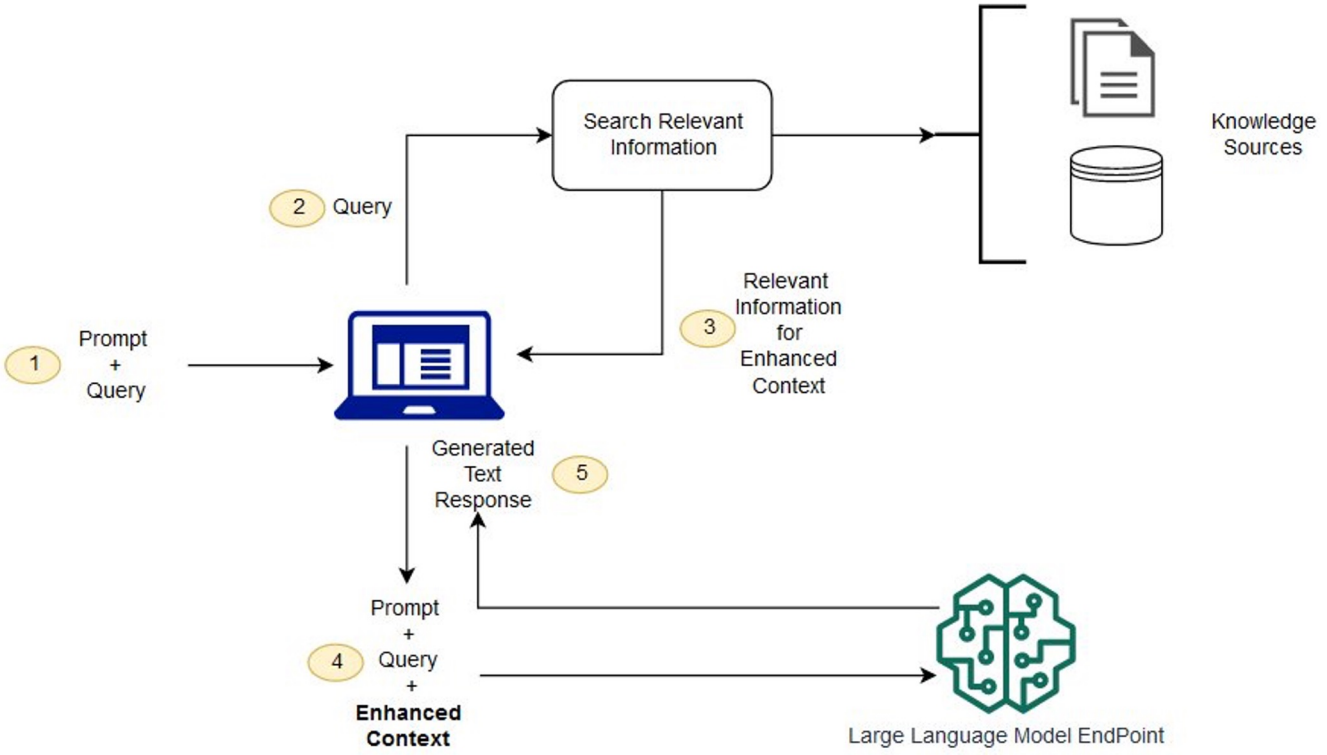
Retrieval Augmented Generation

From RAGs to Riches

Retrieval-Augmented Generation (RAG) is the process of optimizing the output of a large language model, so it references an authoritative knowledge base outside of its training data sources before generating a response.[15]

The benefit of a RAG framework is its ability to enhance AI systems by combining information retrieval, question answering, and text generation capabilities, thereby improving the accuracy and contextuality of responses.

Architecture of a RAG?[16]



DEMO TIME

1. Ask ChatGPT to generate fake restaurant reviews with corresponding ratings
2. Vectorized the reviews using mxbai-embed-large embedding model
3. Feed the reviews back into MongoDB
4. Set up an Atlas Vector Search index
5. Use MongoDB as my Vector Database
6. Write and vectorize prompt/query
7. Retrieve information from MongoDB
8. Pass results from MongoDB as context into the language generation model llama2