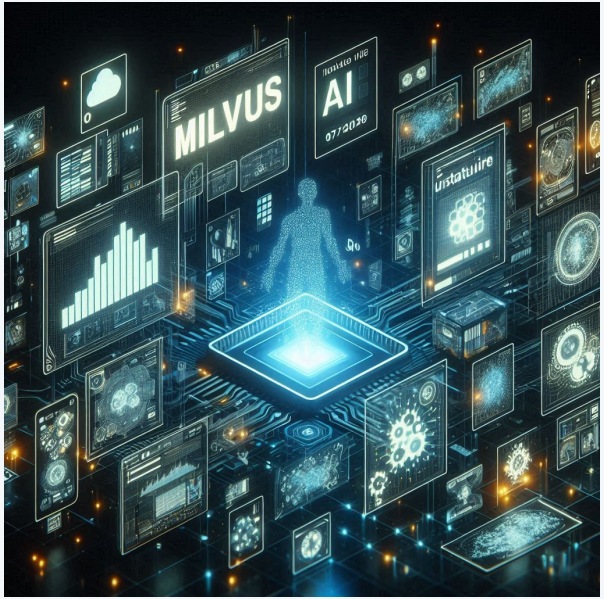
Step-by-Step - RAG 101

Tim Spann @ Zilliz
Principal Developer Advocate

Slides



XX

zilliz

Tim Spann

Principal Developer Advocate, Zilliz

tim.spann@zilliz.com
https://www.linkedin.com/in/timothyspann/
https://x.com/PaaSDev

 |

Agenda

01 About Zilliz, Milvus and Vector Databases

02 Level 100 - Basic RAG

03 References

zilliz

VECTOR DATABASE

zilliz

The Forrester Wave™ Vector Database Providers, Q3 2024

Zilliz is the right partner for your Vector Database needs.

zilliz

Milvus: The most widely-adopted vector database

Milvus is an **Open-Source Vector Database** to **store**, **index**, **manage,** and **use** the massive number of **embedding vectors** generated by deep neural networks and LLMs.

milvus
LF AI &DATA

283+
contributors

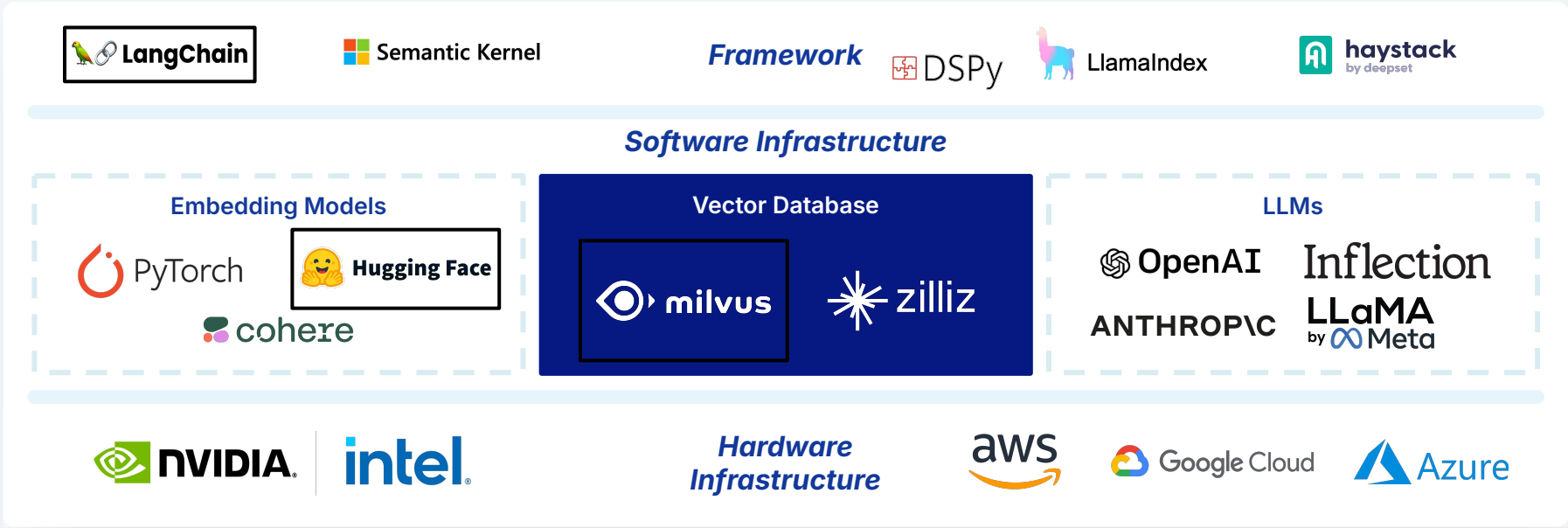
30K+
stars

67M+
docker pulls

2.8K+
forks

zilliz

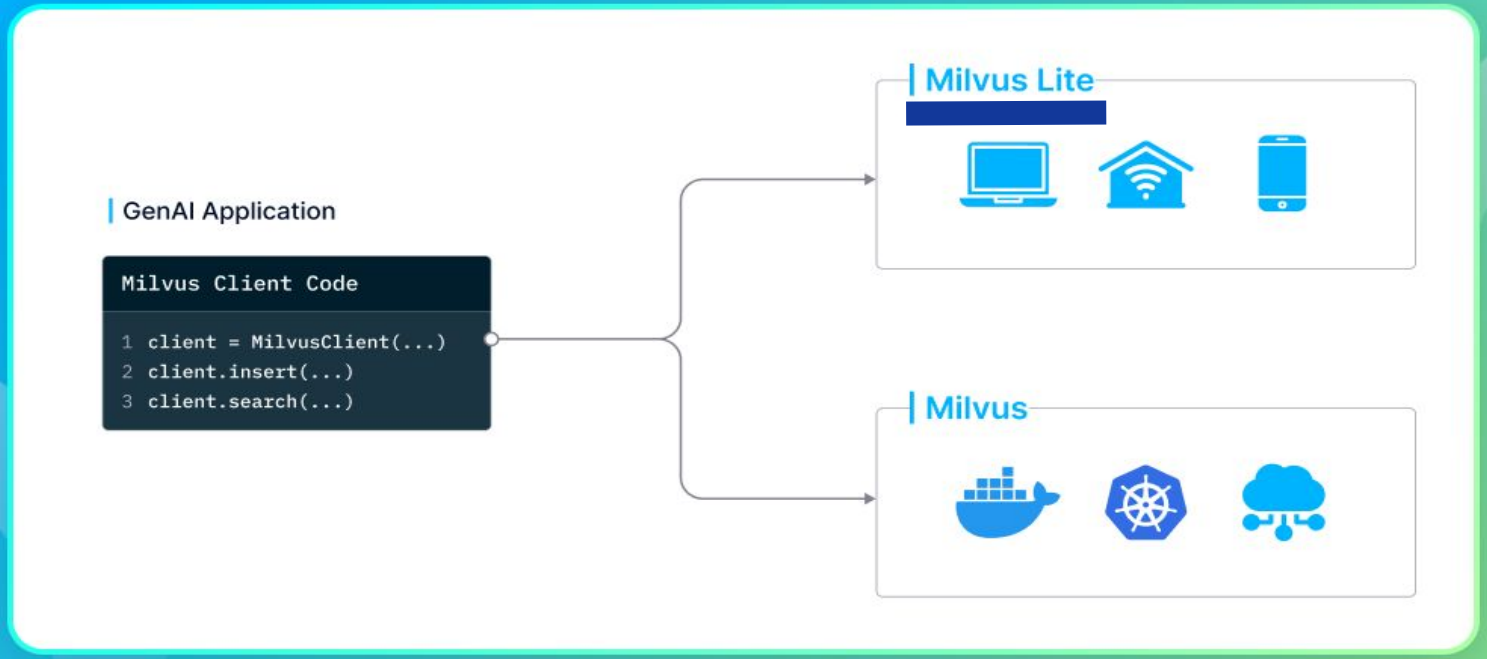
Easy Open RAG Stack Highlighted



LLaMA by Meta

zilliz

Build Once
Deploy Anywhere



milvus

 |

zilliz

Vector Search Overview

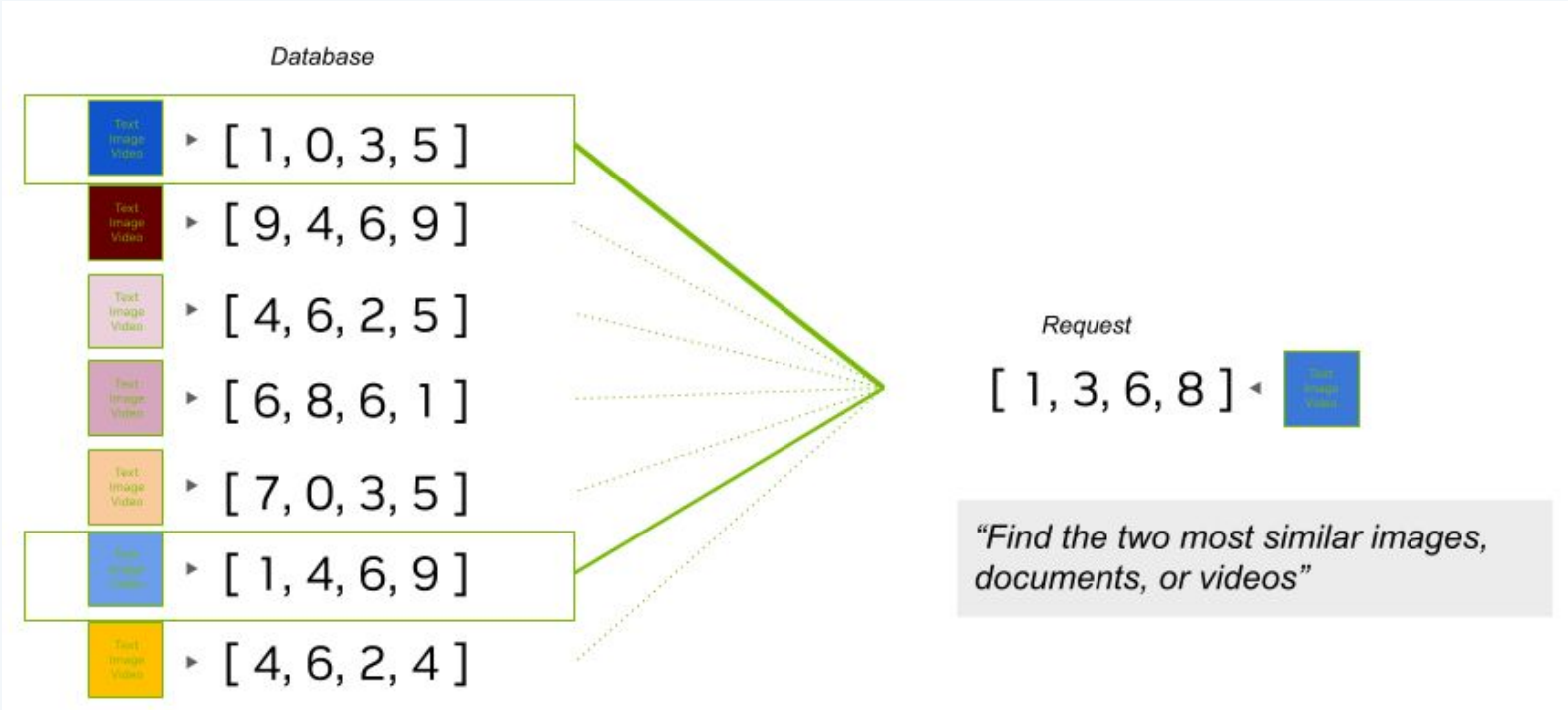
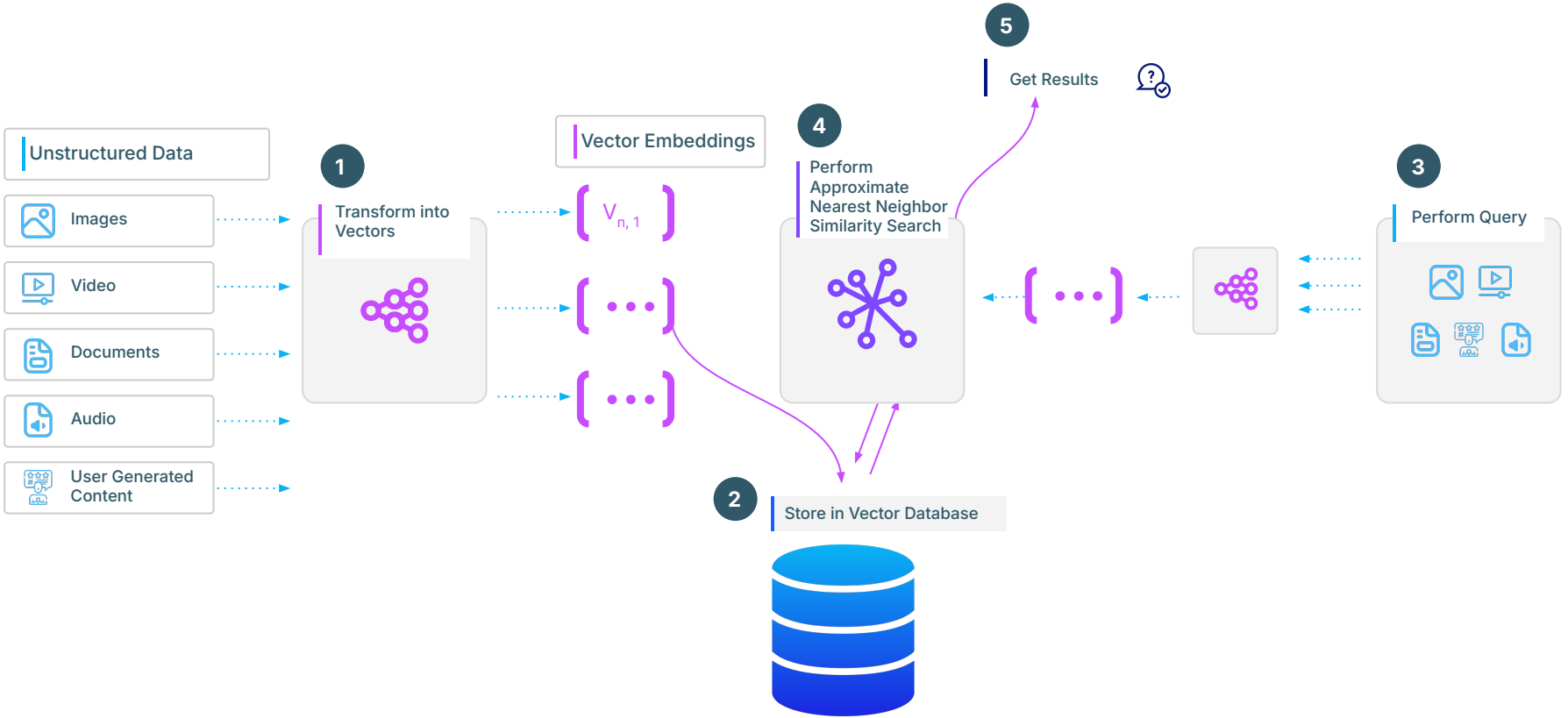


Image from Nvidia

zilliz

How Similarity Search Works



RAG

(Retrieval Augmented Generation)

 |

zilliz

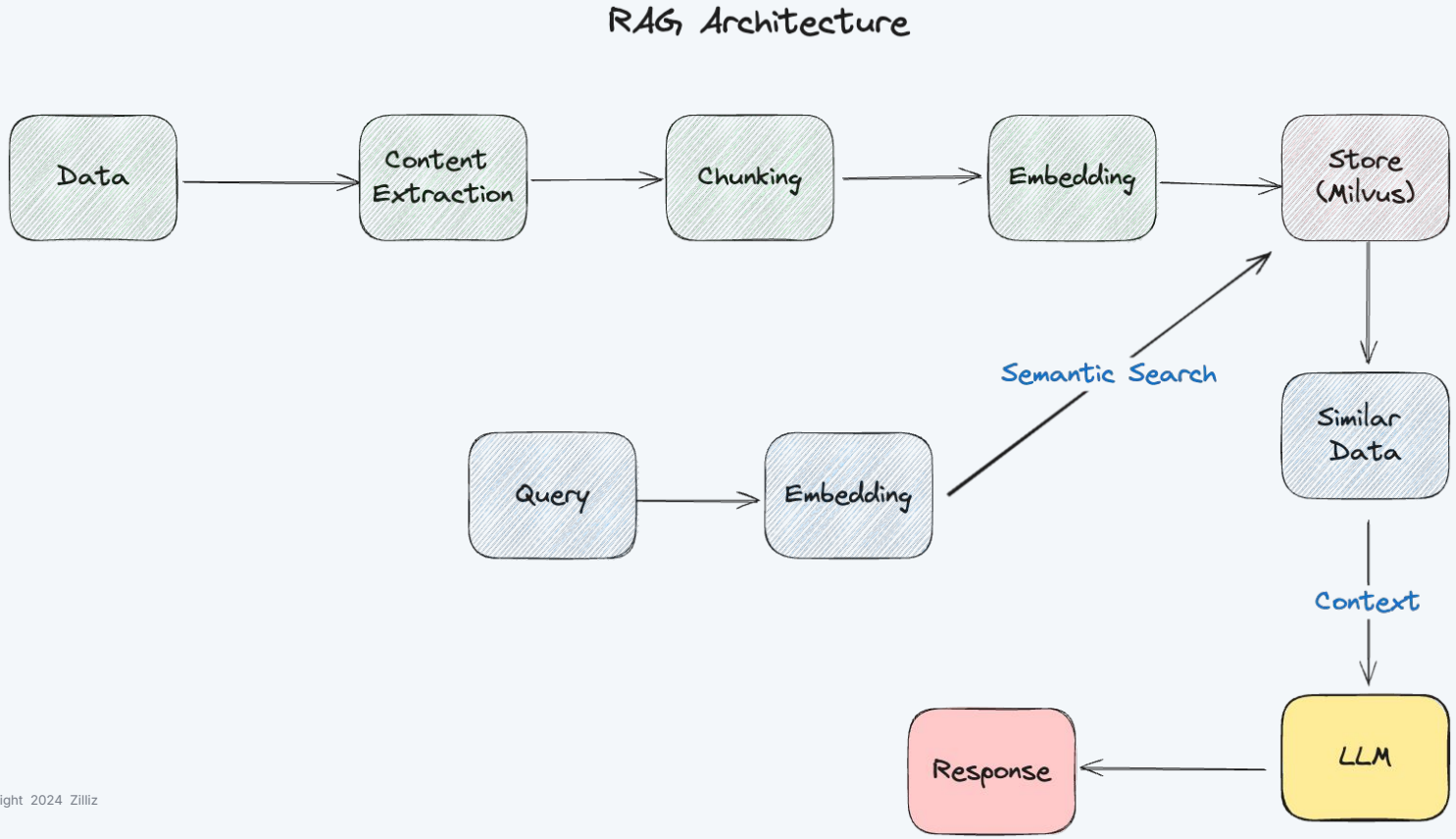
Basic Idea

Use RAG to **force** the **LLM to work with your data** by injecting it via a vector database like **Milvus**

 |

zilliz

Basic RAG Architecture



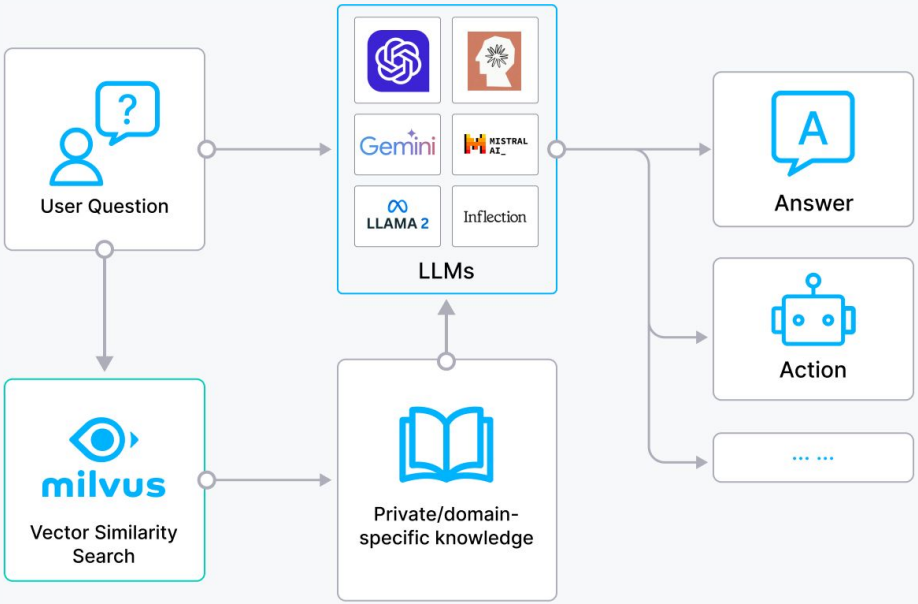
 |

zilliz

Retrieval-Augmented Generation (RAG)

A technique that combines the strength of retrieval-based and generative models:

- Improve accuracy and relevance
- Eliminate hallucination
- Provide domain-specific knowledge

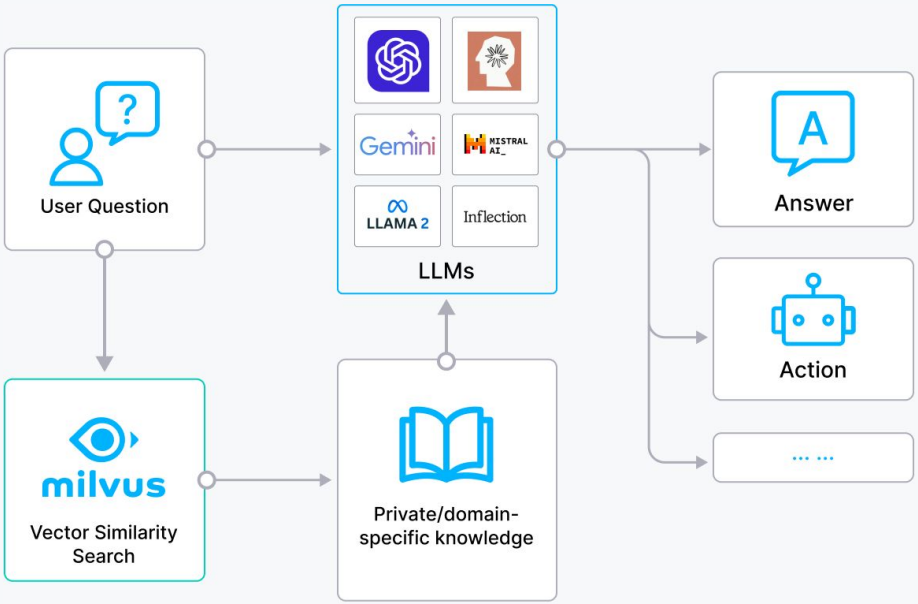


zilliz

RAG : an economic perspective

A business model that bridges public data and private data

- Data sovereignty
- You can't and shouldn't give your private data to others



zilliz

Your Turn

Time To Build a Simple RAG App

zilliz

Documents From Here

https://bit.ly/3U7Okgu

Notebook From Here

https://bit.ly/3BV4IKX

Step 1 - Start off local

Download Python (or use your local install)

https://www.python.org/downloads/

```
python3.11 -m venv yourenv
source yourenv/bin/activate
```

Create an environment

https://docs.python.org/3/library/venv.html

Use Pip

https://pip.pypa.io/en/stable/installation/

Setup a .env file for environment variables

zilliz

Download Jupyter Lab

https://jupyter.org/

Run your notebook

```
jupyter lab --ip="0.0.0.0" --port=8881 --allow-root
```

Running on a Mac or Linux machine is optimal.

Setup environment variables

```
source .env
```

Alternatives

Download Conda

https://docs.conda.io/projects/conda/en/latest/index.html

https://colab.research.google.com/

zilliz

Python SDK Connect...

From:

> *client = MilvusClient(“db/milvus_demo.db”)*

To:

> *client = MilvusClient(uri=”http://server:19530")*

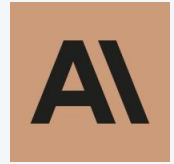
Or:

> *client = MilvusClient(uri=”https://server12345.serverless.gcp-us-west1.cloud.zilliz.com", token=”tokenX”)*

zilliz

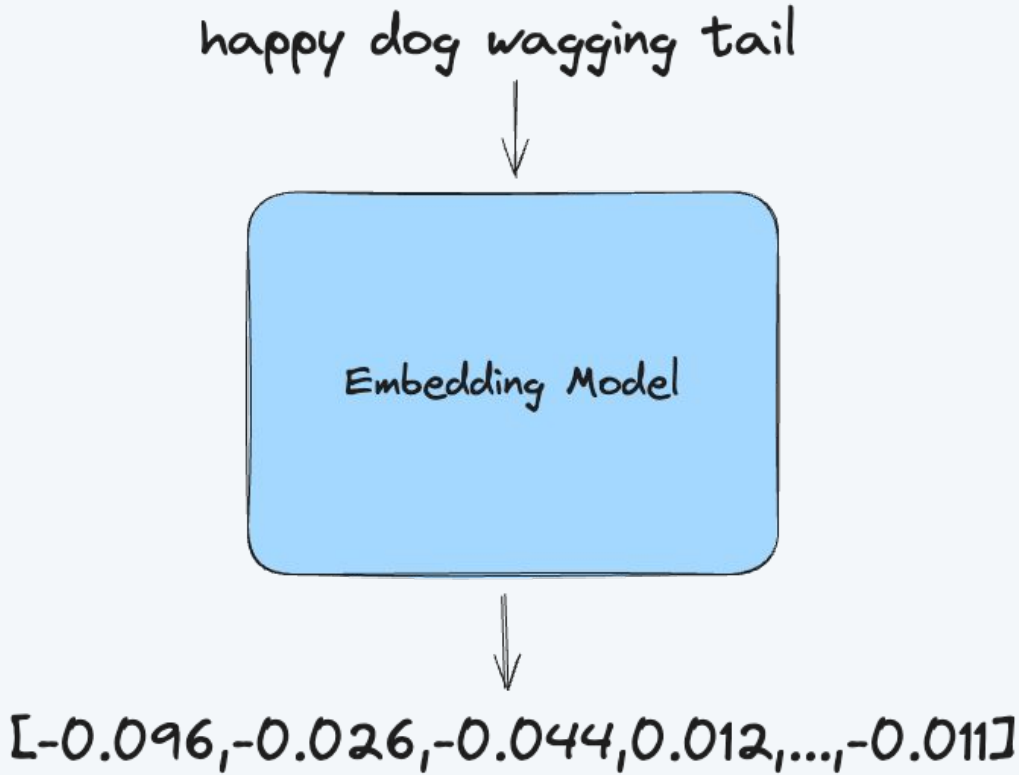
Choose Your Embedding Function

Embedding Function	Type	API or Open-sourced
openai	Dense	API
sentence-transformer	Dense	Open-sourced
bm25	Sparse	Open-sourced
Splade	Sparse	Open-sourced
bge-m3	Hybrid	Open-sourced
voyageai	Dense	API
jina	Dense	API
cohere	Dense	API
Instructor	Dense	Open-sourced
Mistral AI	Dense	API
Nomic	Dense	API
mGTE	Hybrid	Open-sourced



zilliz

Vector Embedding



 |

zilliz

Goal of this Notebook

In this notebook we use langchain to build a simple RAG to Ollama and we ask the llama3 model for data from the slides context fed from Milvus.

🚕 Simple Retrieval-Augmented Generation (RAG) with LangChain:

Build a simple Python RAG application to use Milvus for asking about Tim's slides via OLLAMA. W=

🔍 Summary ¶

By the end of this application, you'll have a comprehensive understanding of using Milvus, data ingest object semi-structured and unstructured data, and using Open Source models to build a robust and efficient data retrieval system.

🐍 AIM Stack - Easy Local Free Open Source RAG

- Ollama
- Python
- Jupyter Notebooks
- Llama 3.2
- Milvus-Lite
- LangChain

 |

zilliz

🔥 Install Ollama

🛰️ Download for Mac, Linux, Windows

https://ollama.com/download

👽 Install Open Source Llama 3.2 model from Meta

https://ollama.com/library/llama3.2

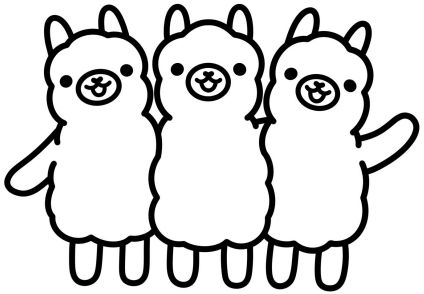
```
ollama run llama3.2

>>> /bye
```

Running the model will download many gigabytes of model and weights for you. When it is complete it will put you interactive chat mode. You can test it, or type */bye* to exit.

🤹 List all of your models

```
ollama list

NAME                       ID              SIZE      MODIFIED
llava:7b                   8dd30f6b0cb1    4.7 GB    40 hours ago
mistral-nemo:latest        994f3b8b7801    7.1 GB    9 days ago
gemma2:2b                  8ccf136fdd52    1.6 GB    10 days ago
nomic-embed-text:latest    0a109f422b47    274 MB    10 days ago
```

Meta



ollama run llama3

zilliz

📊 Let's use it

🦙 First, let's import all the libraries we will need

[2]:
```
import os
from pymilvus import MilvusClient
from langchain_community.document_loaders import PyPDFLoader
from langchain.text_splitter import RecursiveCharacterTextSplitter
from langchain.chains import RetrievalQA
from langchain_milvus import Milvus
from langchain_community.embeddings import HuggingFaceEmbeddings
from langchain.callbacks.manager import CallbackManager
from langchain.callbacks.streaming_stdout import StreamingStdOutCallbackHandler
from langchain import hub

#### Constant For PDF Downloads, If you Change This, Change in Section Below As Well
path_pdfs = "talks/"

#### Initialize Our Documents
documents = []
```

🦪 Download some PDFs of talks

1. Build a directory for the talks
2. Download PDFs

Note: ¶

You can use your own PDFs, download more from

- https://github.com/tspannhw/SpeakerProfile
- https://www.slideshare.net/bunkertor/presentations

[25]:
```
!mkdir talks
!wget -P talks/ https://raw.githubusercontent.com/tspannhw/SpeakerProfile/main/2024/01-oct-2024pes-vectordatabasesandai-241001142959-0510fbe6.pdf
!wget -P talks/ https://raw.githubusercontent.com/tspannhw/SpeakerProfile/main/2024/06-04-2024-nyctechweek-discussiononvectordatabasesunstructureddataandai-240605125759-d80c571c.pdf
!wget -P talks/ https://raw.githubusercontent.com/tspannhw/SpeakerProfile/main/2024/06-04-2024-nyctechweek-localragwithllama3andmilvus-240605113619-6e91032b.pdf
!wget -P talks/ https://raw.githubusercontent.com/tspannhw/SpeakerProfile/main/2024/08-13-2024nycmeetup-unstructureddataprocessingfromcloudtoedge-240812185343-3ae3ff2b.pdf
!wget -P talks/ https://raw.githubusercontent.com/tspannhw/SpeakerProfile/main/2024/09-12-2024milvussensordatarag-240907202906-ea0e6890.pdf
!wget -P talks/ https://raw.githubusercontent.com/tspannhw/SpeakerProfile/main/2024/09-18-2024nycmeetup-vectordatabases102-240917124641-19bae3b0.pdf
```


zilliz

Iterate through PDFs and load into documents

```
for file in os.listdir(path_pdfs):
    if file.endswith(".pdf"):
        pdf_path = os.path.join(path_pdfs, file)
        print(pdf_path)
        loader = PyPDFLoader(pdf_path)
        documents.extend(loader.load())
```

```
talks/09-18-2024nycmeetup-vectordatabases102-240917124641-19bae3b0.pdf
```

Connect to Milvus

Use Milvus-Lite for local database

This is ./milvusrag101.db

You can easily switch to Docker for a more advance Milvus or Zilliz Cloud

You could drop the collection if you are starting new

```
MILVUS_URL = "./rag101.db"

client = MilvusClient(uri=MILVUS_URL)

if client.has_collection("LangChainCollection"):
    print("Collection exists")
else:
    client.drop_collection("LangChainCollection")
```

```
Collection exists
```

 |

zilliz

🐍 AIM Stack - Easy Local Free Open Source RAG

Choose Your Model

https://zilliz.com/ai-models

Free, Hugging Face Hosted, Open Source Apache Licensed

https://zilliz.com/ai-models/all-minilm-l12-v2

```
    embeddings = HuggingFaceEmbeddings(model_name="all-MiniLM-L6-v2")
```

Powerful JINA AI Model for Text Embedding

https://zilliz.com/ai-models/jina-embeddings-v2-base-en

Next step, chunk up our big documents

```
text_splitter = RecursiveCharacterTextSplitter(chunk_size=500, chunk_overlap=100)

all_splits = text_splitter.split_documents(documents)
```

We load our JINA AI text embedding model via HuggingFace

Then we load all of the splits and embeddings to Milvus

This will take some town and will download the model and meta data

```
        drop_old=True
```

That will delete any previously loaded documents. If you set it to False, you don't have to reload your documents. You would be adding more and these could be duplicates.

Verify Documents are Loaded

We run a *similarity_search* on the newly loaded vector store

Reference

https://milvus.io/docs/integrate_with_langchain.md

zilliz

```python
from langchain_huggingface import HuggingFaceEmbeddings
model_kwargs = {"device": "cpu", "trust_remote_code": True}

embeddings = HuggingFaceEmbeddings(model_name="jinaai/jina-embeddings-v2-base-de",  model_kwargs=model_kwargs)

vectorstore = Milvus.from_documents(
    documents=documents,
    embedding=embeddings,
    connection_args={
        "uri": MILVUS_URL,
    },
    drop_old=False,
)

vectorstore.similarity_search("What is Milvus?", k=1)
```

```
[Document(metadata={'page': 23, 'pk': 453229992137722003, 'source': 'talks/06-20-2024-aicampmeetup-unstructureddataandvectordatabases-240620160248-1964efbc.pdf'}, page_content='24\n|   © Copyright Zilliz 24\nMilvus: From Dev to Prod \nAI Powered Search made easy \nMilvus is an Open-Source Vector \nDatabase  to store, index, manage, and \nuse the massive number of embedding  \nvectors  generated by deep neural \nnetworks and LLMs. \ncontributors \n267+\nstars\n27K+\ndownloads \n25M+\nforks\n2K+\n')]
```


zilliz

Code our loop to call LLama 3.2

We will pull the RAG prompt information from LLama's hug and connect the documents loaded into Milvus with our LLM chat with LLama 3.2

```python
from langchain_ollama import OllamaLLM

def run_query() -> None:
    llm = OllamaLLM(
        model="llama3.2",
        callback_manager=CallbackManager([StreamingStdOutCallbackHandler()]),
        stop=["<|eot_id|>"],
    )

    query = input("\nQuery: ")
    prompt = hub.pull("rlm/rag-prompt")

    qa_chain = RetrievalQA.from_chain_type(
        llm, retriever=vectorstore.as_retriever(), chain_type_kwargs={"prompt": prompt}
    )

    result = qa_chain.invoke({"query": query})
    # print(result)
```

```python
if __name__ == "__main__":
    while True:
        run_query()
```

```
Query:  What indexes are available in Milvus?
I don't know what indexes are available in Milvus, as the provided context only mentions that it is an Open-Source Vector Database without specifying the types of indexes. The context als
o does not provide any information about indexing capabilities or options.

Query:  What is Milvus?
Milvus is an Open-Source Vector Database used to store, index, manage, and use the massive number of embedding vectors generated by deep neural networks and LLMs. It provides an AI-powere
d search solution made easy. Milvus is developed in a collaborative environment with over 267 contributors.

Query:  What is a vector database?
A vector database is a type of data storage system designed for efficient storage and retrieval of dense vectors, often used in applications such as recommendation systems and computer vi
sion. It stores and indexes large amounts of numerical data, allowing for fast querying and similarity searches. Vector databases are particularly useful for tasks that require computing
distances or similarities between high-dimensional vectors.
```


zilliz

zilliz

Metadata Filtering in LangChain
with Milvus Vector Database

Step 1: Install the LangChain wrapper for Milvus langchain-milvus and other dependencies.

```
$ pip install --upgrade --quiet  langchain langchain-core langchain-community langchain-text-splitters langchain-milvus langchain-openai bs4
```

Step 2: Ingest data to Milvus through the Milvus.from_documents() abstraction.

```
from langchain_milvus import Milvus from langchain_openai import OpenAIEmbeddings  embeddings = OpenAIEmbeddings()  vectorstore = Milvus.from_documents(  # or Zilliz.from_documents
documents=docs,     embedding=embeddings,     connection_args={
"uri": "./milvus_demo.db",  # or network endpoint for Milvus server.
},     drop_old=True,  # Drop the old Milvus collection if it exists
)
```

Step 3: Perform semantic similarity search and apply filter expr to only select from a certain publishing source.

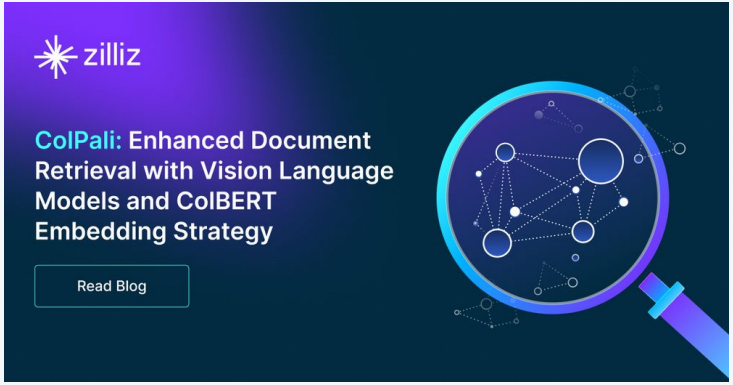
```
vectorstore.similarity_search(

"What is CoT?",

k=1,

expr="source == 'https://lilianweng.github.io/posts/2023-06-23-agent/'",

)
```

 |

zilliz

Advanced RAG

zilliz



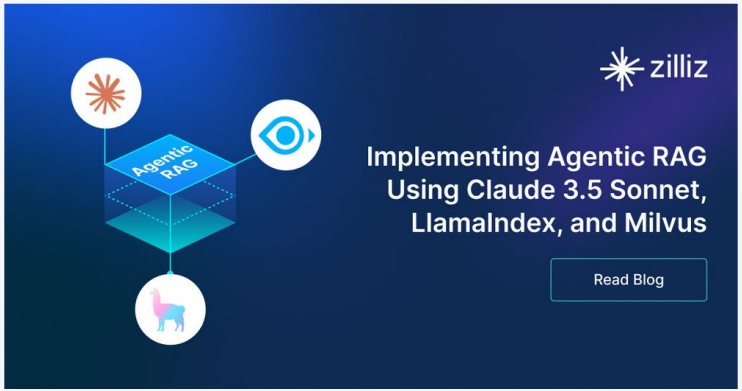
https://bit.ly/4eFdMlK

https://bit.ly/3BLeLCx

 |

zilliz

https://bit.ly/3zXW8dX



https://bit.ly/3NuK5ru

zilliz

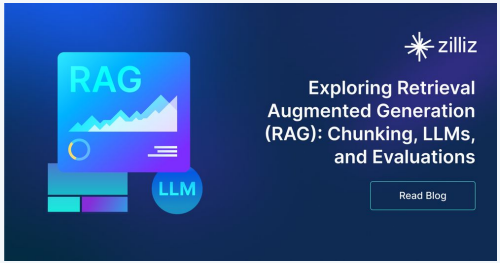
https://bit.ly/4gZ4Lpn

Metadata Filtering

Hybrid Search

Agents

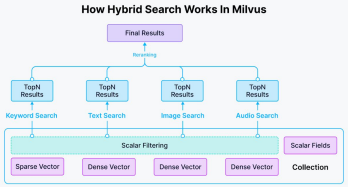
https://bit.ly/3UbqUqx



https://bit.ly/3YpKd1K

Smart Chunking

Embedding Model Choice



 |

zilliz

Resources

zilliz

Wednesday, October 23, 2024
5:30 PM to 8:30 PM EDT NYC
https://bit.ly/403JIvy

startupgrind

Supercharging Startup
Vector Databases, and
Oct 24, 6:30 – 8:00 PM (EDT)
Princeton
Startup Grind Princeton, 23 Orchard Road, Montgomery, 08558

Thursday, October 24, 2024
6:30 PM to 8:00 PM EDT Princeton
https://bit.ly/3NopXY2

November 5-7, 10-12, 2024: CloudX
https://www.developerweek.com/cloudx/

November 15, 2024: Build Stuff
https://www.developerweek.com/cloudx/speakers/

November 19, 2024: Xtreme Python
https://xtremepython.dev/2024/

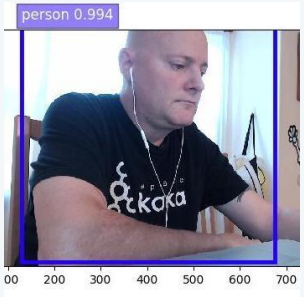
November 21, 2024: bigdata Conference

November 21, 2024: Unstructured Data NYC

zilliz

Vector Database Resources

Give Milvus a Star!

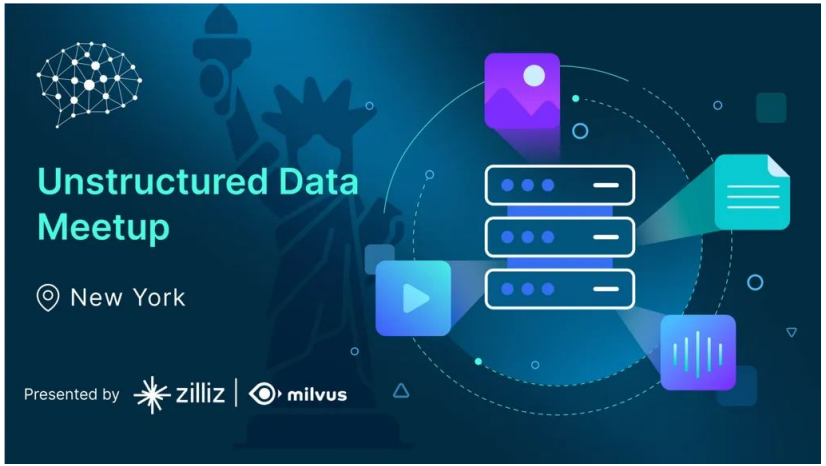


Chat with me on Discord!

https://github.com/milvus-io/milvus

zilliz

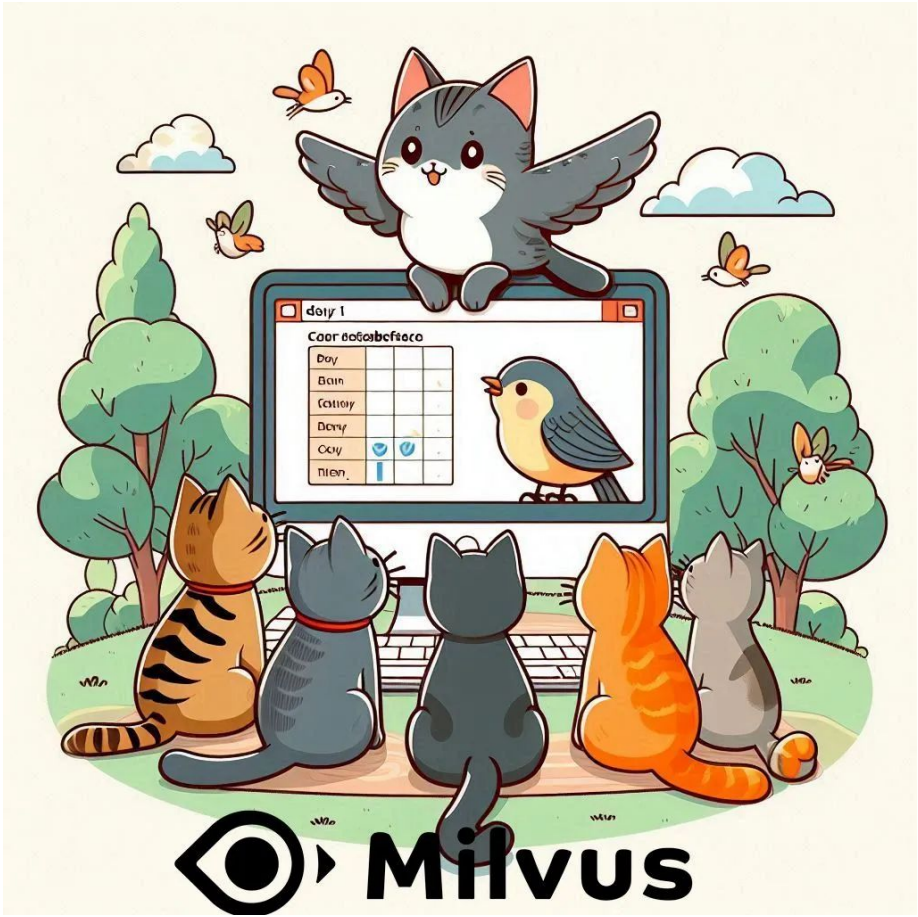
Unstructured Data Meetup



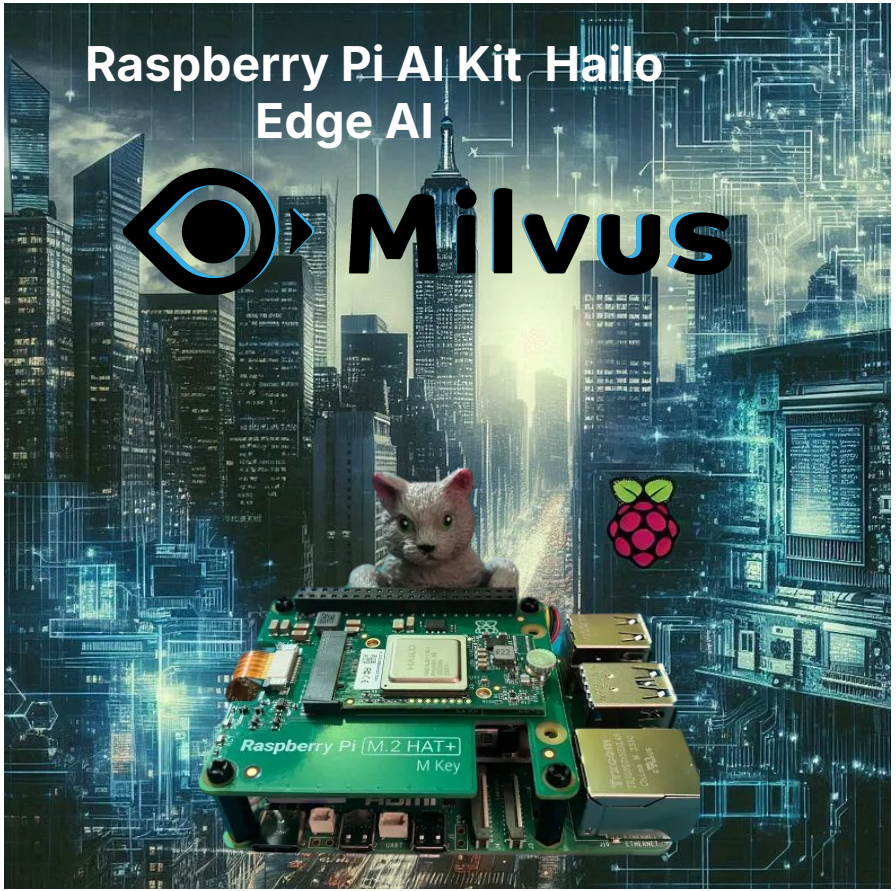
https://www.meetup.com/unstructured-data-meetup-new-york/

This meetup is for people working in unstructured data. Speakers will come present about related topics such as vector databases, LLMs, and managing data at scale. The intended audience of this group includes roles like machine learning engineers, data scientists, data engineers, software engineers, and PMs.

This meetup was formerly Milvus Meetup, and is sponsored by Zilliz maintainers of Milvus.



https://medium.com/@tspann/shining-some-light-on-the-new-milvus-lite-5a0565eb5dd9



https://medium.com/@tspann/unstructured-data-processing-with-a-raspberry-pi-ai-kit-c959dd7fff47

AIM Weekly by Tim Spann

https://bit.ly/32dAJft
https://github.com/milvus-io/milvus

This week in Milvus, Towhee, Attu, GPT Cache, Gen AI, LLM, Apache NiFi, Apache Flink, Apache Kafka, ML, AI, Apache Spark, Apache Iceberg, Python, Java, Vector DB and Open Source friends.

 |

zilliz

Thank you!

milvus.io

github.com/milvus-io/

@milvusio

Connect with me!

@paasDev

/in/timothyspann

 |

zilliz

Getting Started with Vector Databases

Milvus

Open Source Self-Managed

github.com/milvus-io/milvus

29K+ - Star us on GitHub!

Zilliz Cloud

SaaS Fully-Managed

zilliz.com/cloud

 |

zilliz

zilliz

Get started for free

zilliz.com/cloud

Deep-Dive

 |

zilliz

Vector Similarity Measures: L2 (Euclidean)

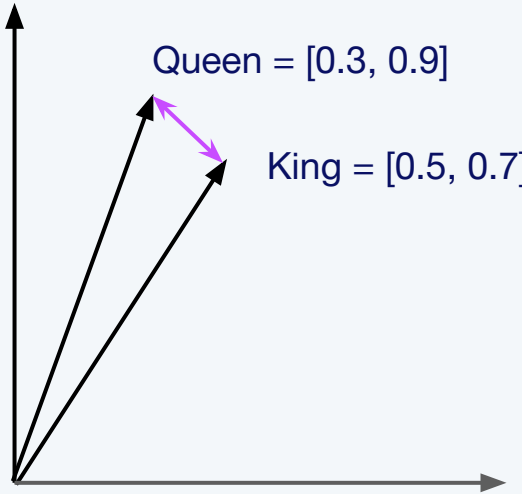
$$d(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

d(Queen, King) = $\sqrt{(0.3-0.5)^2 + (0.9-0.7)^2}$

= $\sqrt{(0.2)^2 + (0.2)^2}$

= $\sqrt{0.04 + 0.04}$

= $\sqrt{0.08} \cong 0.28$



 |

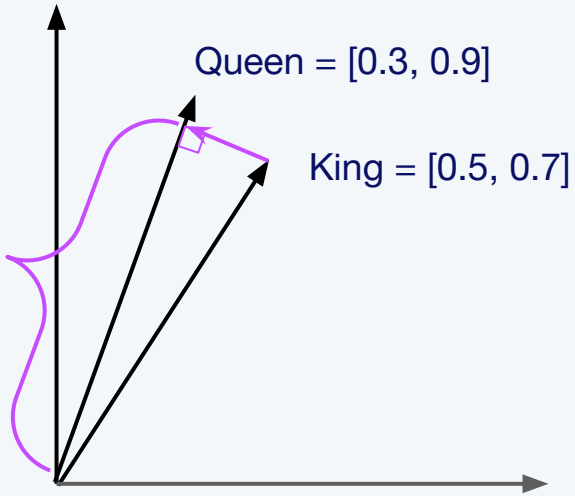
zilliz

Vector Similarity Measures: Inner Product (IP)

$$a \cdot b = \sum_{i=1}^{n} a_i b_i$$

Queen · King = (0.3*0.5) + (0.9*0.7)

= 0.15 + 0.63 = 0.78

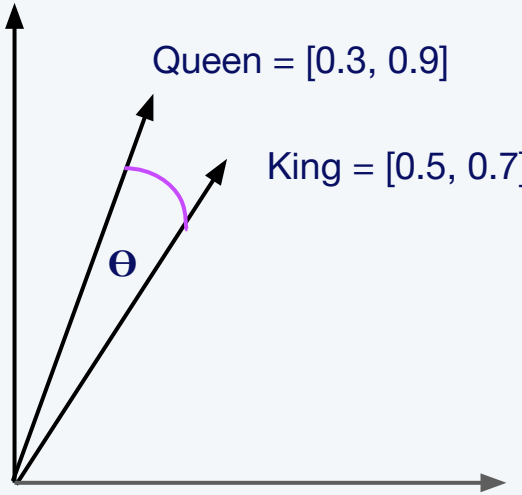


 |

zilliz

Vector Similarity Measures: Cosine

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$



cos(Queen, King) = $\frac{(0.3*0.5)+(0.9*0.7)}{\sqrt{0.3^2+0.9^2} * \sqrt{0.5^2+0.7^2}}$

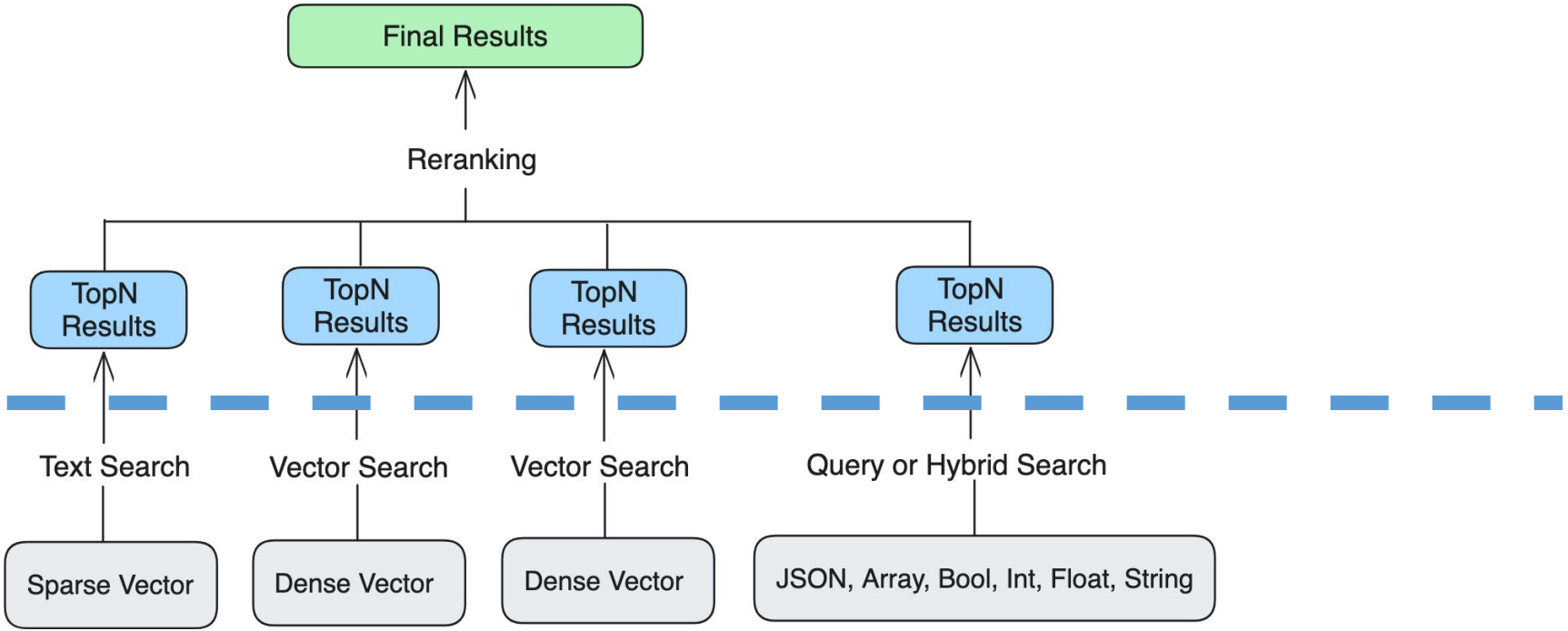
= $\frac{0.15+0.63}{\sqrt{0.9} * \sqrt{0.74}}$

= $\frac{0.78}{\sqrt{0.666}}$

≅ 0.03

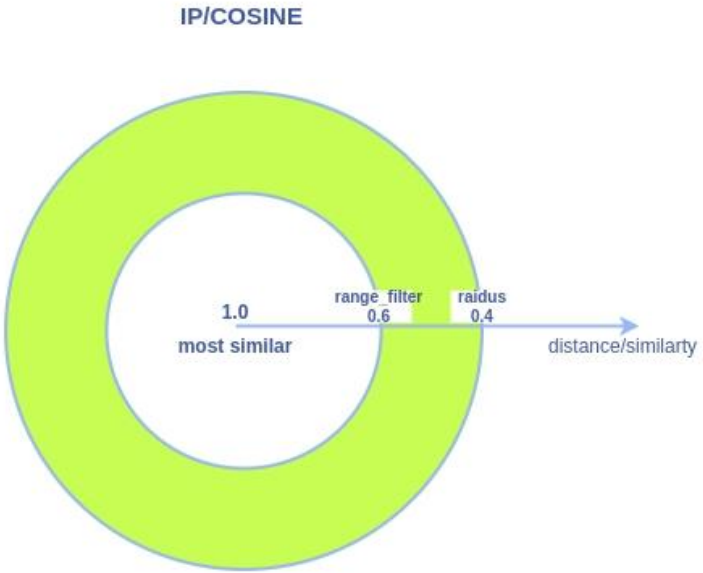
zilliz

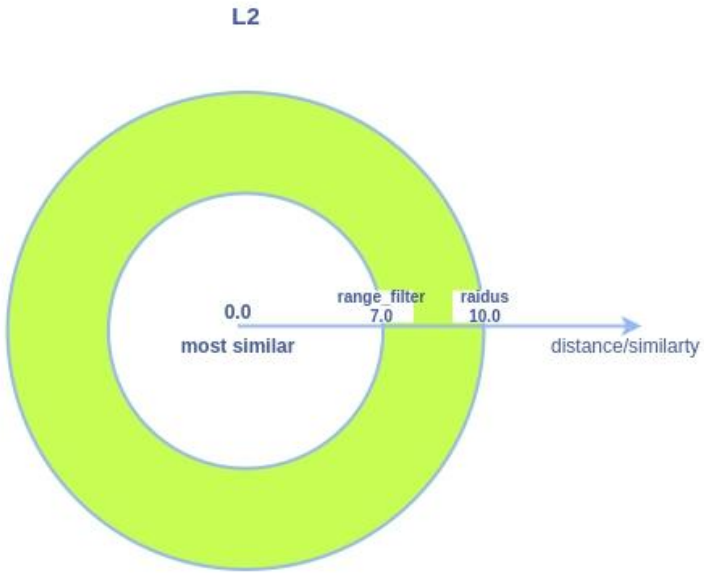
Hybrid Search



 |

zilliz






```
search_params = { "metric_type": "COSINE", "params": { "radius": 0.4, "range_filter":0.6} }
```

```
search_params = { "metric_type": "L2", "params": { "radius": 10.0, "range_filter":7.0} }
```


zilliz