

ARI Hands-on Session

Retrieval-Augmented Generation

*Minu Mathew
Research Software Engineer
National Center for Supercomputing Applications*



NCSA

import MinuMathew as mm

mm.meet()

mm.work()



Research Software Engineer



Adjunct Lecturer



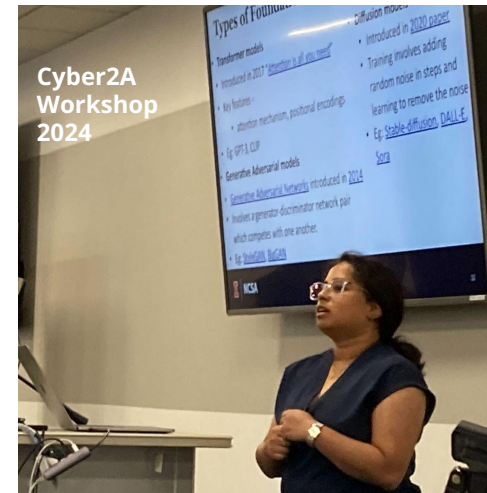
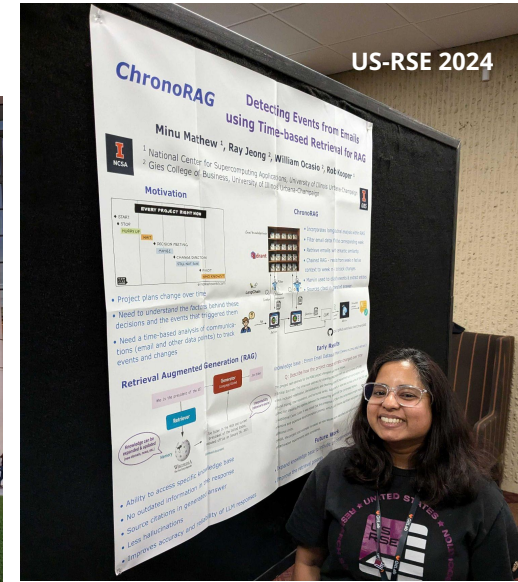
Machine Learning Intern



Research Student



Software Engineer



mm.fun()



NCSA

Agenda

- Motivation
- Intro
- UIUC.chat
- RAG explained
- Coding!



Slides and Code:

<https://github.com/minump/HandsOn-RAG/tree/main>



Motivation

- LLM - Large pre-trained Language Models :



Fine-tuning LLM. Source: [datacamp blog](https://datacamp.com/blog/fine-tuning-llm)

- No ability to access a specific knowledge
 - LLM has not seen “your” data.
 - If specific domain knowledge is required - fine tune.

Motivation

- LLMs :
 - No sources
 - Responding without context
 - Hallucinations
 - Out-of-date information

Explain to me what an llm is

An LLM, or Master of Laws, is a postgraduate degree in law that is typically pursued by individuals who have already completed a bachelor's degree in law (LLB or JD) or have significant legal experience. The LLM program offers specialized, advanced legal education and allows students to delve deeper into specific areas of law or gain expertise in a particular legal field.



LLM response without context



When did France gift Lithuania Vilnius TV tower?



France gifted Lithuania the Vilnius TV tower in 1980.

What prompted the French government to provide the gift?

The French government provided the gift as a gesture of friendship and solidarity with the Lithuanian people, who had recently declared independence from the Soviet Union.

LLM Hallucinations

Introduction



- RAG (Retrieval-Augmented Generation)
 - Introduced in 2020 ([paper](#))
 - Generation - by LLMs
 - Retrieval-Augmented
 - Retrieve required information from provided knowledge base
 - Provide this information to LLMs as context to answer user query

Retrieval-Augmented Generation

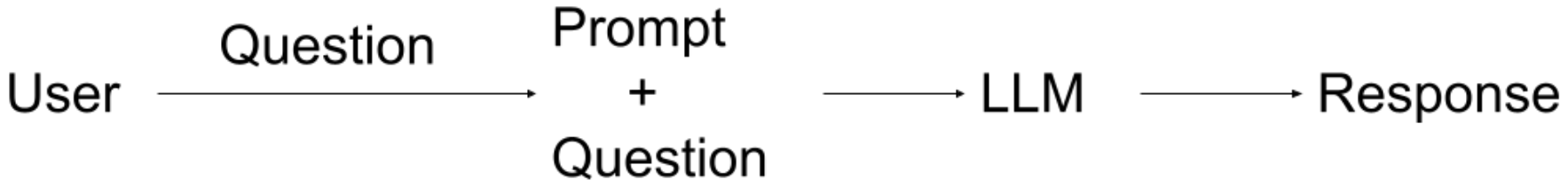
Retrieval

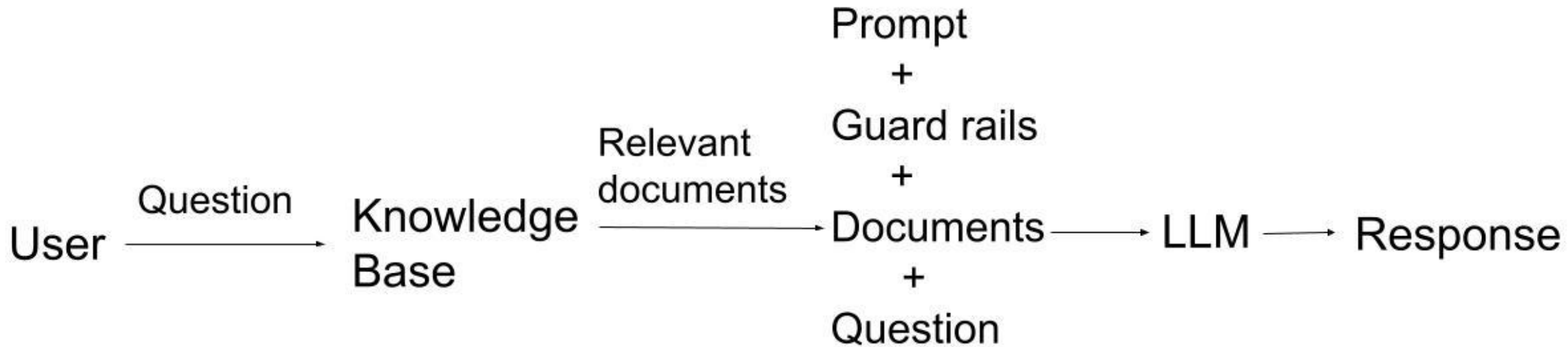
- Set up a knowledge base
- Retrieve documents relevant to the user query

Generation

- Using LLMs
- Use the retrieved documents as context

Retrieval-Augmented Generation

- Before :


```
graph LR; User -- Question --> Prompt_Q[Prompt + Question]; Prompt_Q --> LLM; LLM --> Response
```
- After :


```
graph LR; User -- Question --> KB[Knowledge Base]; KB -- Relevant documents --> Documents; Documents --> Prompt_Guard[Prompt + Guard rails + Documents + Question]; Prompt_Guard --> LLM; LLM --> Response
```


Without RAG

- No ability to access a specific knowledge/domain
- No sources
- Hallucinations
- Out-of-date information

With RAG

- Point to a knowledge base
- Sources cited in LLM response
- LLM response is grounded by relevant information from knowledge base
- Update the knowledge base with new information

UIUC.chat - RAG in Action

- UIUC.chat (<https://www.uiuc.chat/>)
 - Login/Sign up with registered email id
 - Public projects - ECE120, NCSA, NCSA Delta, Clowder...
 - Project - RAGHandsOn (private project page, with OpenAI key)
 - <https://www.uiuc.chat/RAGHandsOn/chat> - chat with LLM
 - <https://www.uiuc.chat/RAGHandsOn/materials> - upload your documents

Check out uiuc.chat for 15 min

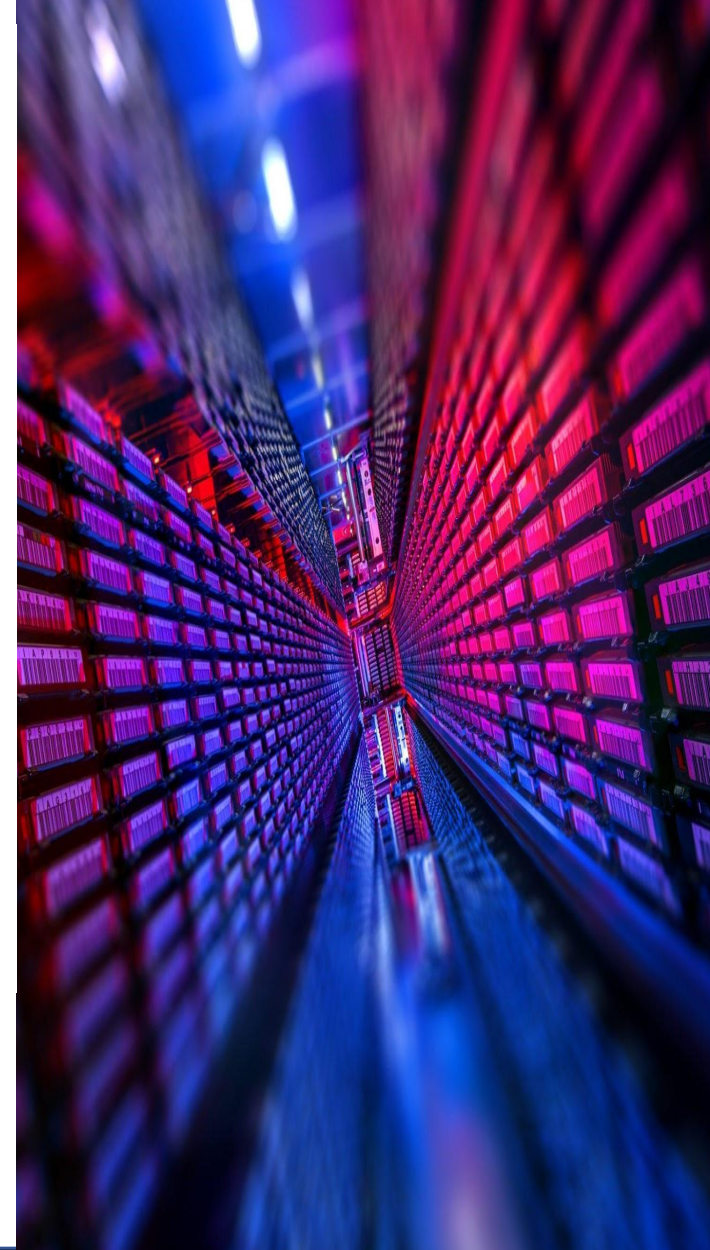
Coding Time!

1. Go to this sheet
2. Claim your user account
 - a. Enter your name and email address



Github <https://github.com/minump/HandsOn-RAG/tree/main>

JupyterHub : <https://rag.class.ncsa.illinois.edu/>



Coding Time!

Prerequisites

- Python
- Using [Langchain](#) framework

Github <https://github.com/minump/HandsOn-RAG/tree/main>

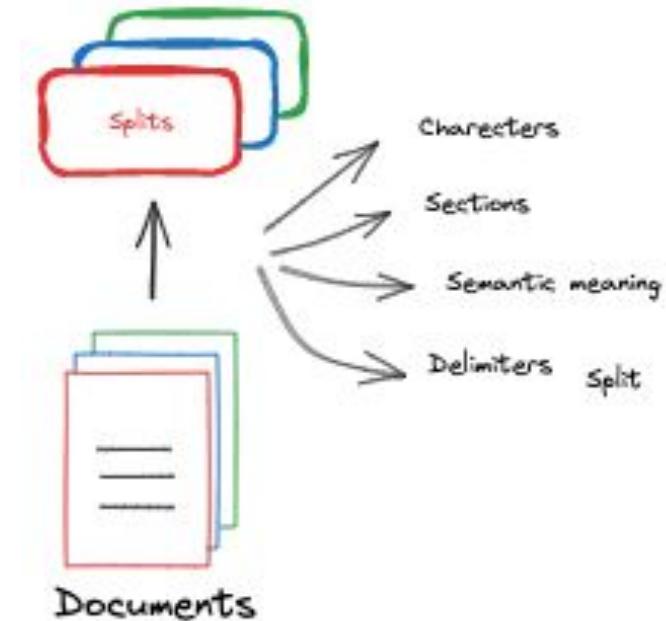
JupyterHub : <https://rag.class.ncsa.illinois.edu/>



RAG - Retrieval-Augmented Generation

Chunking

- Each document contains vast amount of information
- Chunking breaks large amounts of data into smaller, more manageable pieces.
 - LLMs have a limited context window and can't take in the entire dataset at once.
 - For [GPT-4 128k token limit](#).
- Try to split meaningfully
 - by pages, then by sections, paragraphs, sentences, characters..



RAG - Retrieval-Augmented Generation

Vector embeddings

- Representing language in a neural network

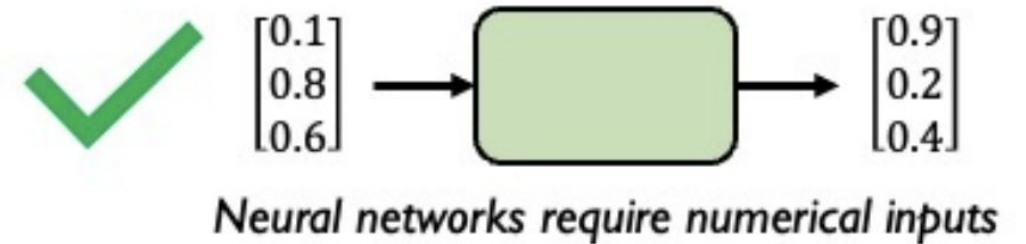
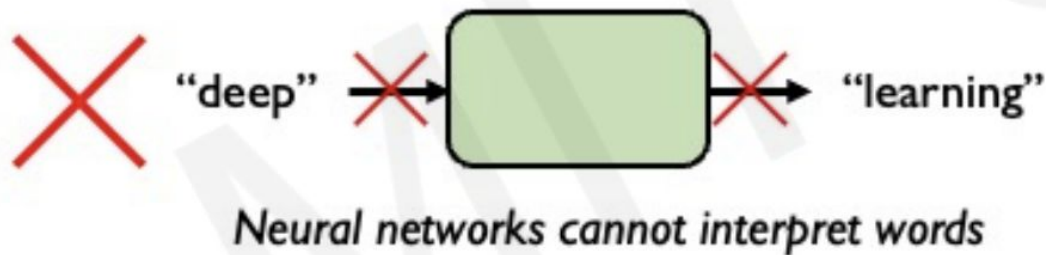
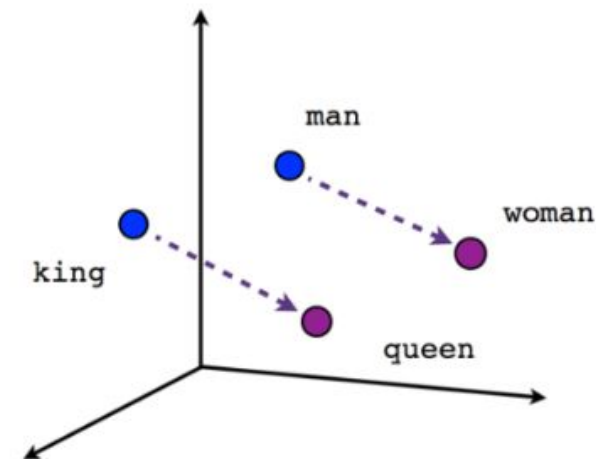


Image source: MIT Deep Learning course [slides](#)

RAG - Retrieval-Augmented Generation

Vector embeddings

- Embeddings - vector representation of text.
- Individual words are represented as real-valued vectors.
- Captures semantic meaning and relationships of the text in a high-dimensional space.
- Words that have similar meaning have similar representation.
- Eg : [GloVE](#), [MTEB leaderboard](#) in HF.
- We will be using OpenAI text embedding model with 8191 vector dimension.

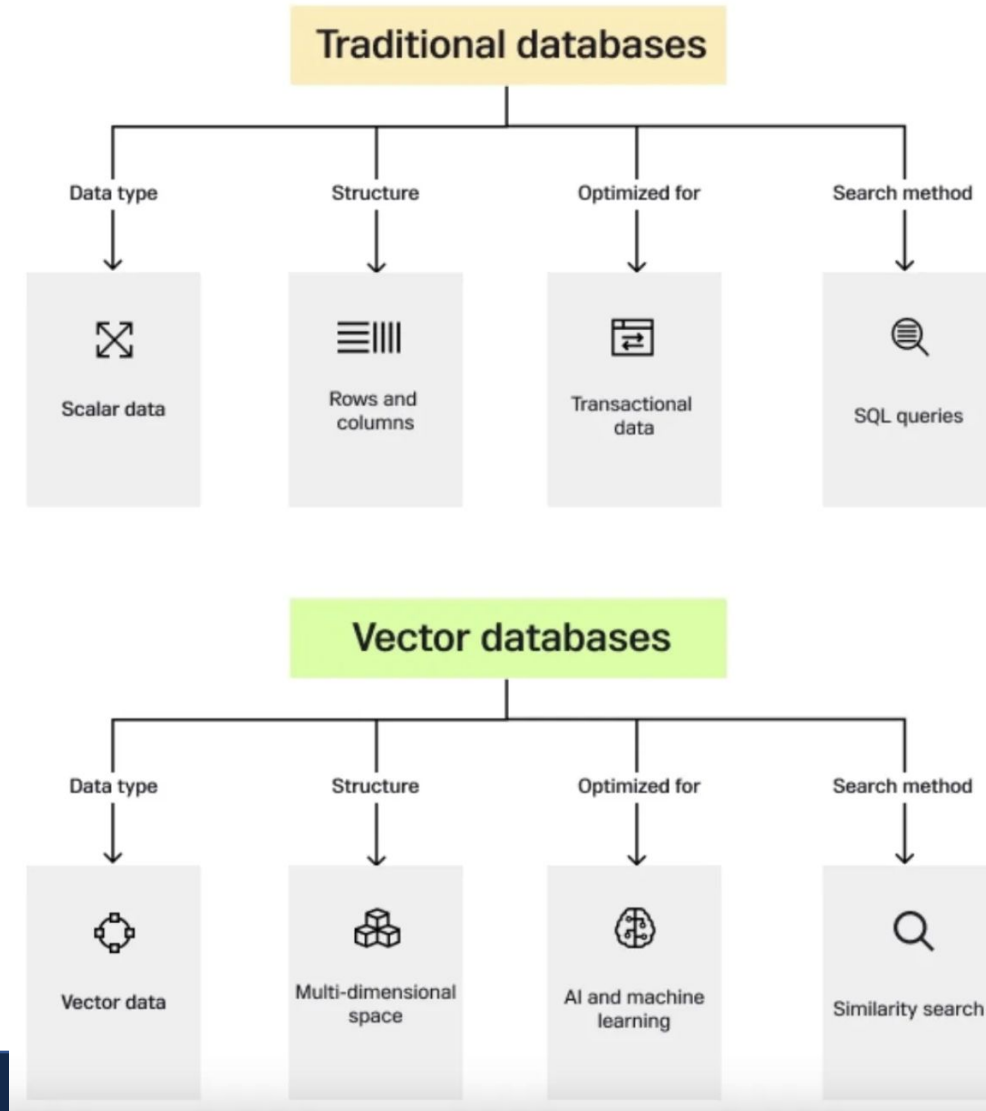


RAG - Retrieval-Augmented Generation

Knowledge DB

- Vector database (Beginners [blog 1](#), Pinecone [blog 2](#))
- Efficiently store, index and search high-dimensional data
- Store data as vector embeddings
- Optimized for fast retrieval and similarity search
- Calculate the distance between user query embedding and other data points

Image source: VectorDB - a beginners guide [blog](#)

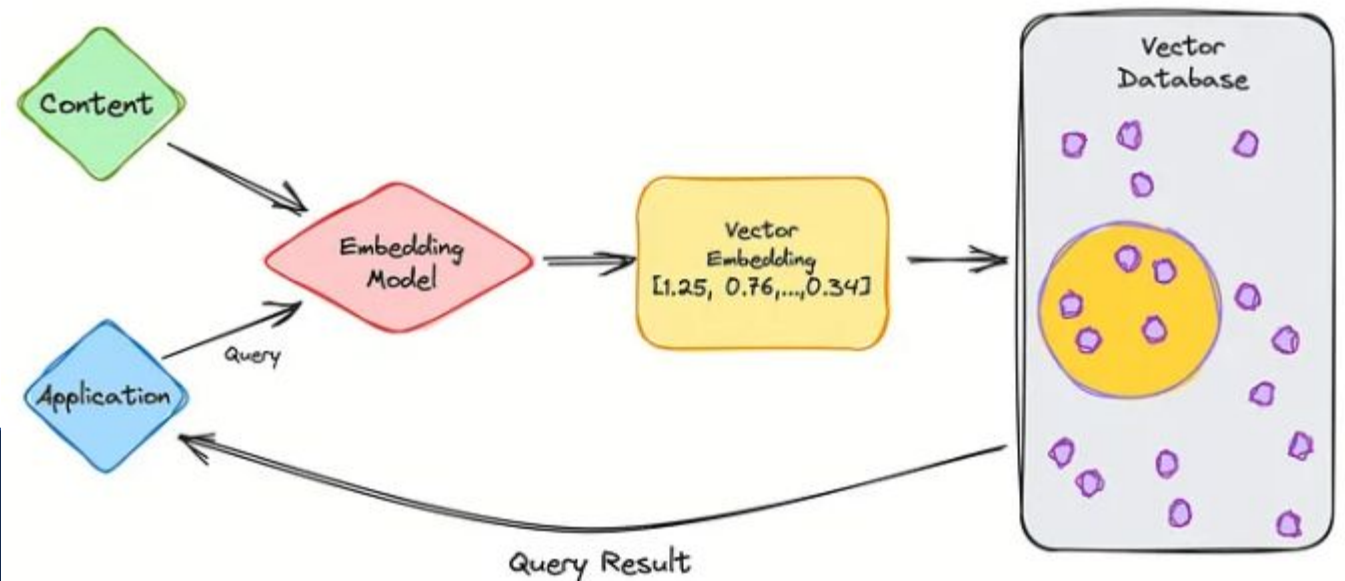


RAG - Retrieval-Augmented Generation

Vector DB Retrieval

1. Partition data into chunks
2. Use embedding model to create vector embeddings for the data (create indexes)
3. Insert vector embeddings into the db, with some reference to the original content/metadata
4. User query - use same embedding model to create user query embedding
5. Query the db for similar vector embeddings
6. Return similar document chunks.

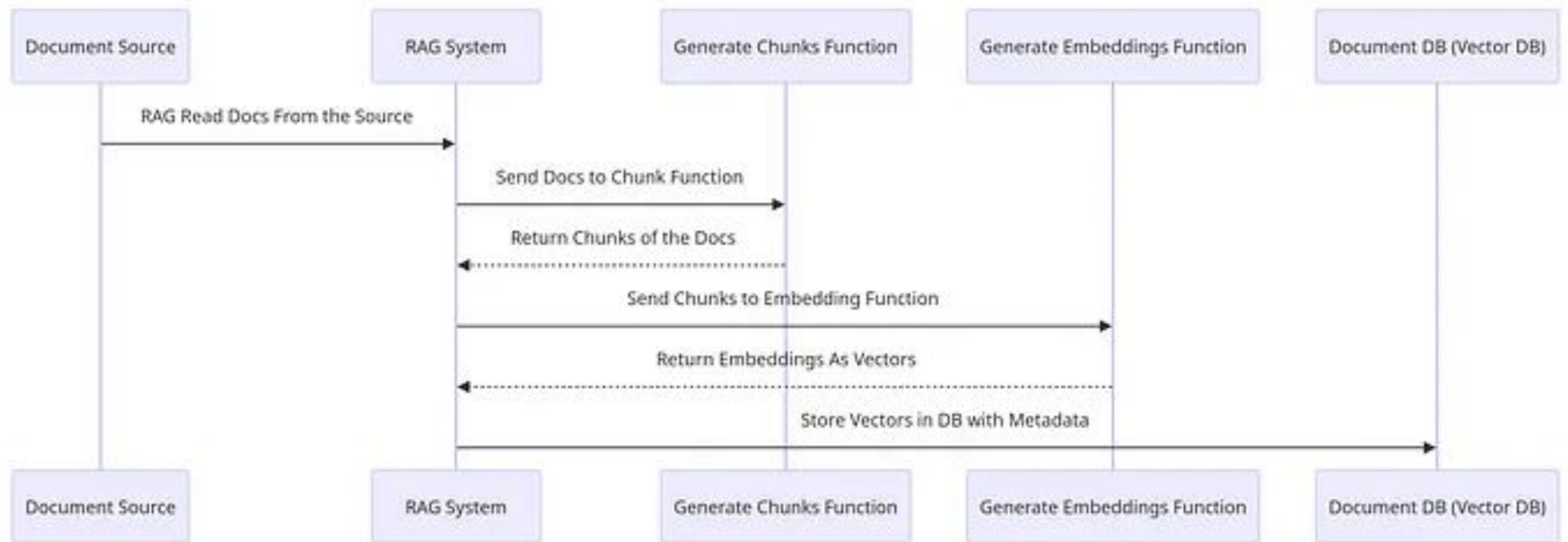
Image Credits : [KDnuggets](#)



RAG - Retrieval-Augmented Generation

Inserting into DB

1. Query the database
2. Retrieve relevant information

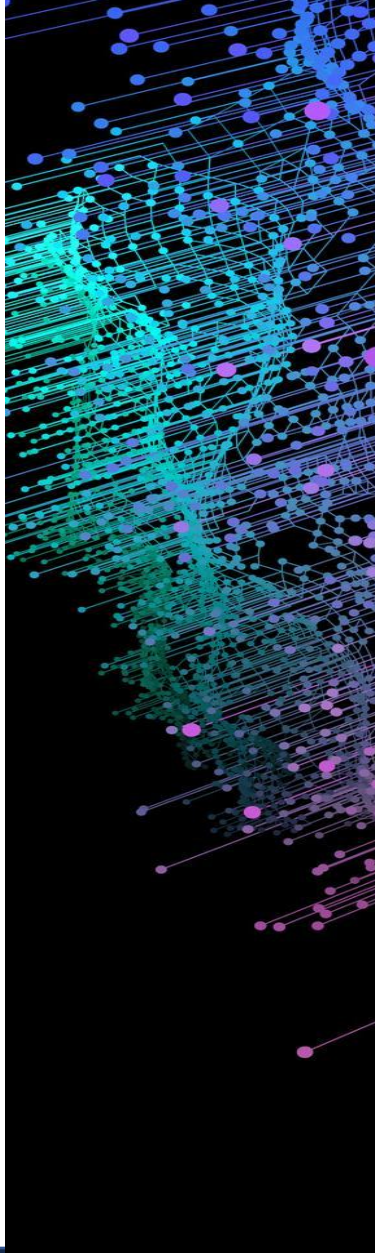


Source Credits :
[Blog.demir](https://blog.demir.dev)

RAG - Retrieval-Augmented Generation

LLM

- Pre-trained transformer models
- Trained to predict the next word (token), given some input text.
- Open-source models - [HuggingFace leaderboard](#)
- For HandsOn - [OpenAI GPT-4o-mini](#), and [Ollama](#) Llama3



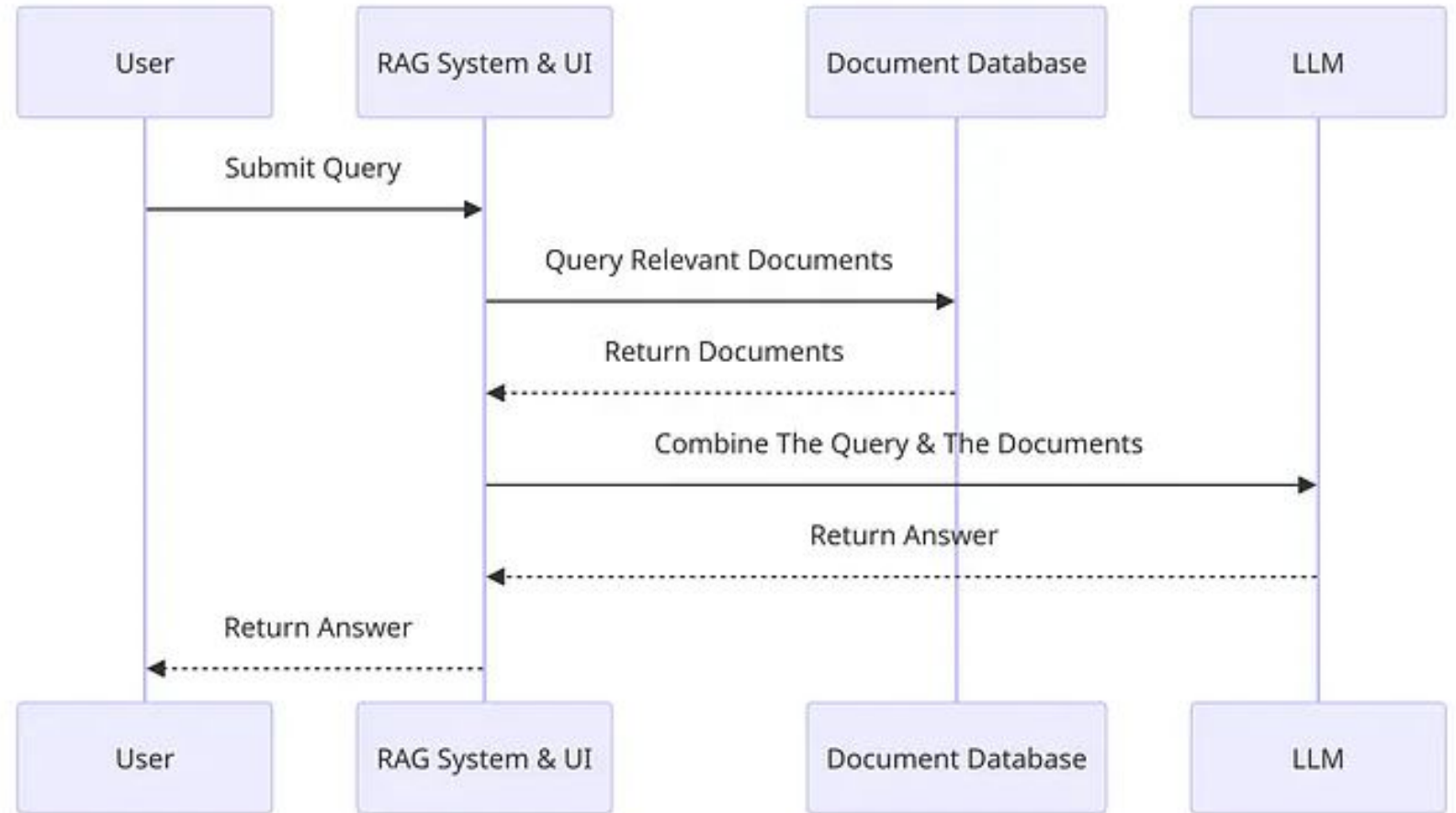
RAG - Retrieval-Augmented Generation

Prompting

- Communicate with LLMs to get desired outcomes without updating the model
- Takes multiple trial-and-errors to get desired effect.
- Include specific persona / behaviour
 - Eg: “You are a helpful research assistant”
- Include guard rails
 - Eg: “If you don’t know the final answer, just say “I don’t know”
- Include instructions
 - Eg: “Read the data file before answering any questions”
- Include response formats
 - Eg: “Respond using markdowns”
- LilianWeng [blog post](#), [medium blog post](#) on prompt engineering

RAG - Retrieval-Augmented Generation

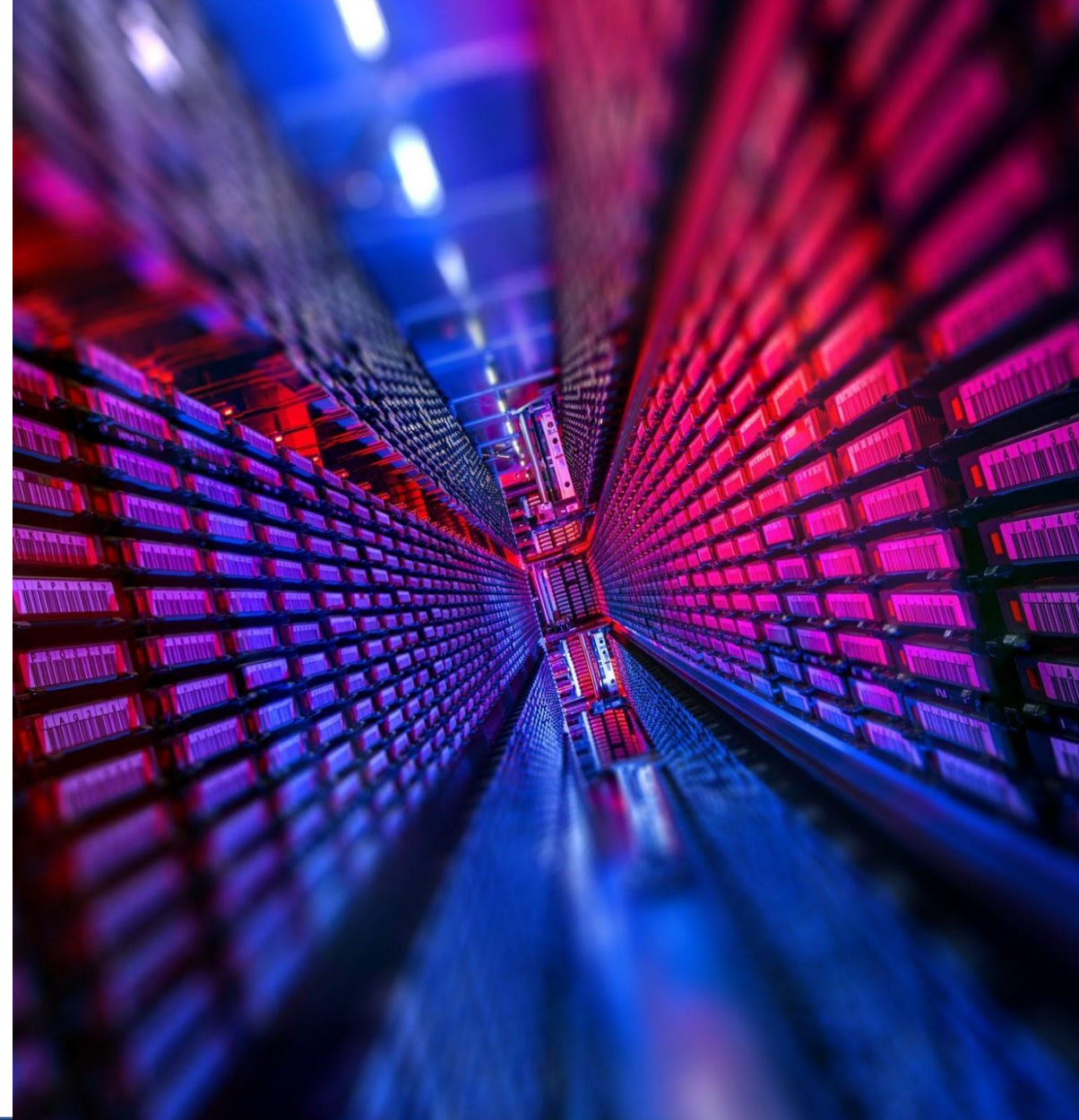
1. Query the database
2. Retrieve relevant information
3. Pass relevant documents + user query to LLM
4. Pass LLM response to user



Source Credits : [Blog.demir](https://blog.demir.dev)

RAG - Hands-on

- Github
- <https://github.com/minump/HandsOn-RAG>
- Jupyter hub
- <https://rag.class.ncsa.illinois.edu/>
- UIUC chat
- <https://www.uiuc.chat/RAGHandsON/chat>





That's all folks!

Minu Mathew

minum@illinois.edu

www.linkedin.com/in/minumpmathews