BUILDING INTELLIGENT ASSISTANTS WITH LLMs

Introduction to Retrieval-Augmented Generation (RAG) with Custom Data

Nottingham DS&AI Meetup
Juan Jimenez, March 2025

BIO

I work as a Senior Software Engineer mainly with Python I did my Master's in Computer Science at the University of Birmingham. I'm a certified Google Cloud Professional Developer. My main focus is on large-scale system design and Machine Learning workflows. I'm currently working at ADMIOS, collaborating as a member of the ML team at Profitmind.

AGENDA

- Introduction to LLMs and RAG
- Picking custom data for RAG
- Integrating custom data with LangChain and OpenAI
- Demo: Building an Intelligent Assistant with RAG
- Q&A Session

Introduction to LLMs and RAG

What is a Foundation model?

- AI Model trained with vast amount of data.
- Very general that serves as foundation for specific use cases.
- Can handle different data types including video, text, code (which is a special case of text), photos, CSV, and so on.

Introduction to LLMs and RAG

What is Adaptation?

- Taking a foundation model and retrain it on specific data.
- Customize to be more accurate.
- Specialized tasks.
- Comply with organizational directives.
- Use updated information.

Prompt: Initial text or instruction provided to a large language model to guide the response generation, influencing both the relevance and accuracy of the model's response.

Introduction to LLMs and RAG

LLM Adaptation Techniques

We will focus on Retrieval-Augmented Generation (RAGs), that is a technique that enhances LLM responses by integrating an external retrieval system. Instead of relying solely on pre-trained knowledge, the model retrieves relevant documents or data from a knowledge base before generating a response, improving accuracy, relevance, and factual consistency.

Other adaptation techniques include:

- Fine-tuning on domain-specific datasets.
- Prompt engineering to optimize model outputs.
- Transfer learning for adapting knowledge from one task to another.
- Parameter-efficient tuning (e.g., LoRA, adapters) to modify behaviour without retraining the full model.

Introduction to LLMs and RAG

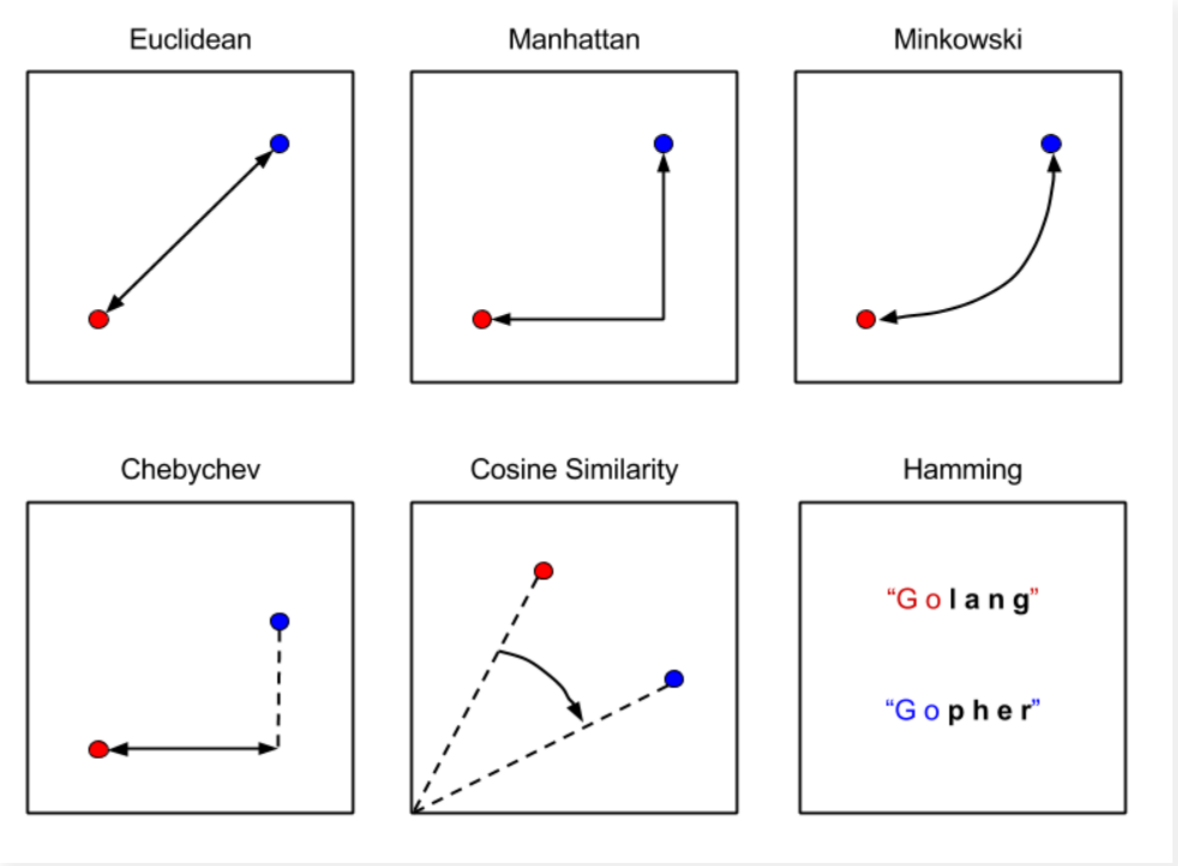
Deep dive into RAG

- The retrieval system fetches specific data pieces that are relevant to the question, and send it as part of the prompt.
- The dataset that contains the relevant information should be updated continuously, with domain specific recent data.
- Syncing strategy is always hard.
- How can we measure "relevance"? (Next slide)
- Multiple chunks could be provided with relevant data.
- Feels like cheating.

Introduction to LLMs and RAG

Relevance measurement in RAG

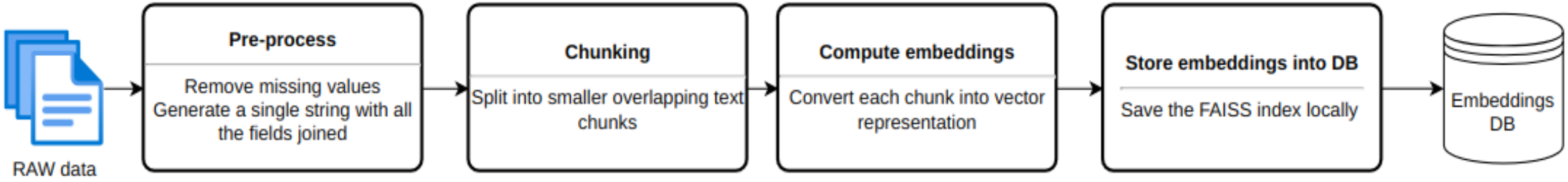
- To measure the similarity between two entities, we will use vectors, as they provide concepts we can use to operate with them: add, subtract, and multiply.
- Usually used the cosine similarity for semantic similarity.
- More about vector similarity here.



Source: Mukherjee, T. (2023). *Different types of distances used in Machine Learning – Explained*. Medium. Link

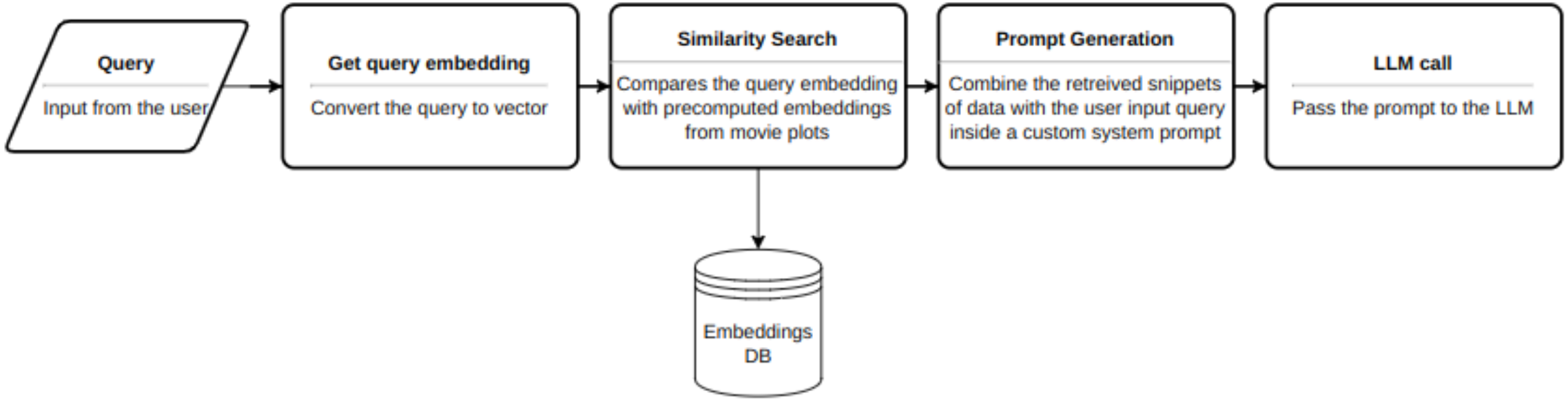
RAG Flows

- Data loading flow.



RAG Flows

- Query flow.



Picking custom data for RAG

- Where to get the data? In this case, it is from Kaggle.
- Original size 81.19 MB.
- Why stats matter?
  - Compute costs beforehand.
  - Estimate storage.
  - Pick the best technology.
  - What data we have to clean?
  - What are the fields that we want to include in the indexing?
  - Pick a significant sample for development purposes.

Cast Main actors/actresses	Genre Movie genre(s)	Wiki Page URL of Wikipedia page from which the plot description was scraped	Plot Long form description of the movie's plot
[null] 4% Tom and Jerry 0% Other (33384) 96%	unknown 17% drama 17% Other (22839) 65%	34070 unique values	33869 unique values

# Release Year Year in which the movie was released	Title Movie title	Origin/Ethnicity Origin of movie (i.e. American, Bollywood, Tamil, etc.)	Director Director(s)
 1901 2017	32432 unique values	American 50% British 11% Other (13839) 40%	Unknown 3% Michael Curtiz 0% Other (33683) 97%

Integrating custom data with LangChain and OpenAI

What is LangChain?

Framework for connecting and using components, under a unified syntax

- LLMs
- Data sources
- Prompts
- Other functionalities

Integrating custom data with LangChain and OpenAI

Why LangChain?

We can decide not to use frameworks for more control, but....

- Manually load and preprocess data
- Chunk data and create embeddings using different libraries
- Manage a separate DB for embeddings
- Implement similarity search
- Build the prompt
- Handle LLM integration
- What happens if we want to change the LLM provider?

Demo: Building an Intelligent Assistant with RAG

- Jupyter notebook: localhost
- GitHub repository:
  - Current slides
  - Vector similarity search slides
  - Code

  https://github.com/rojoyin/ds-ai-meetup-llm-rag-demo

Q&A Session

OUTRO

LET'S CONNECT

GitHub Repository

https://www.linkedin.com/in/juan-carlos-jimenez-illescas

https://github.com/rojoyin

https://stackoverflow.com/users/5682752/juan-carlos-jimenez