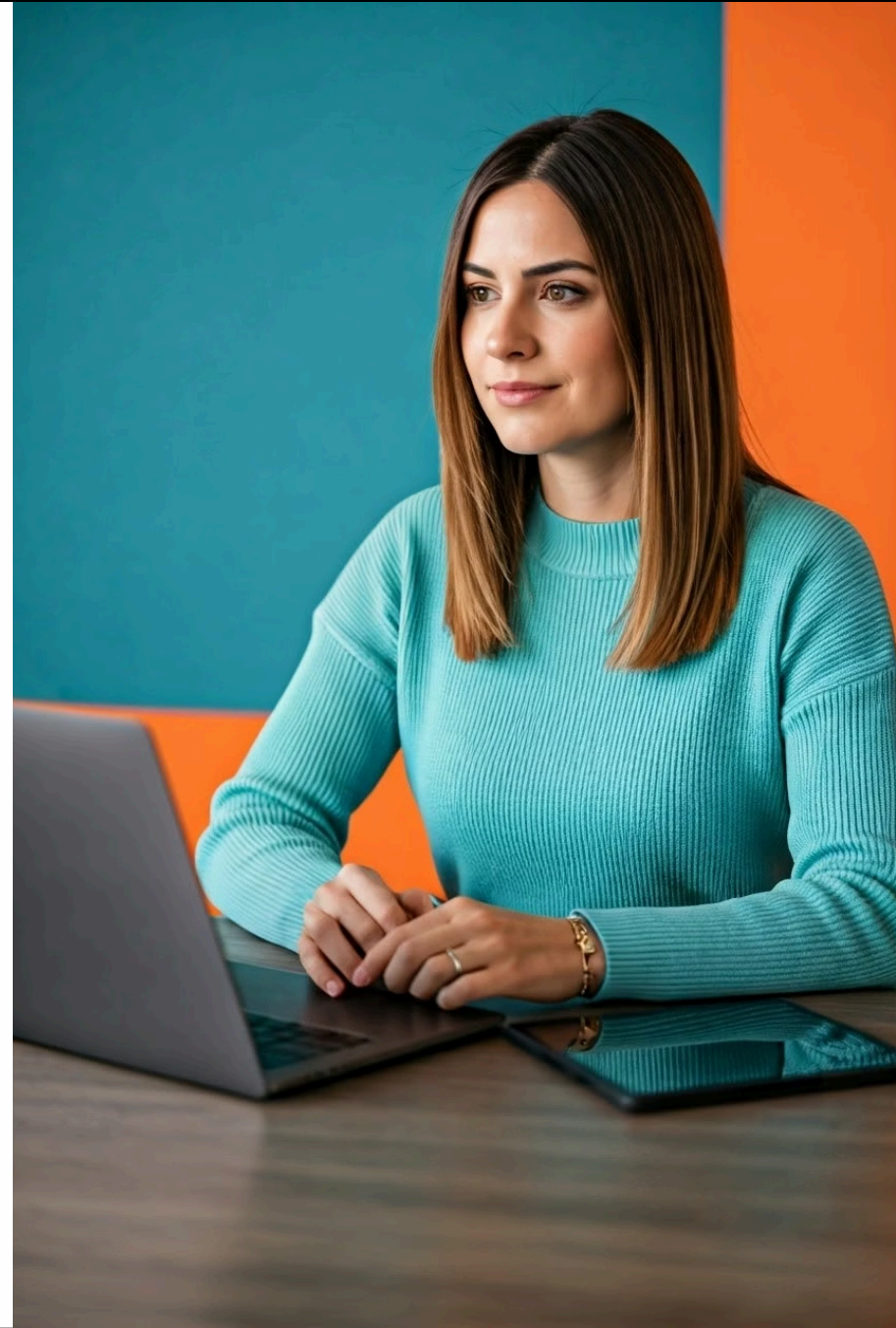
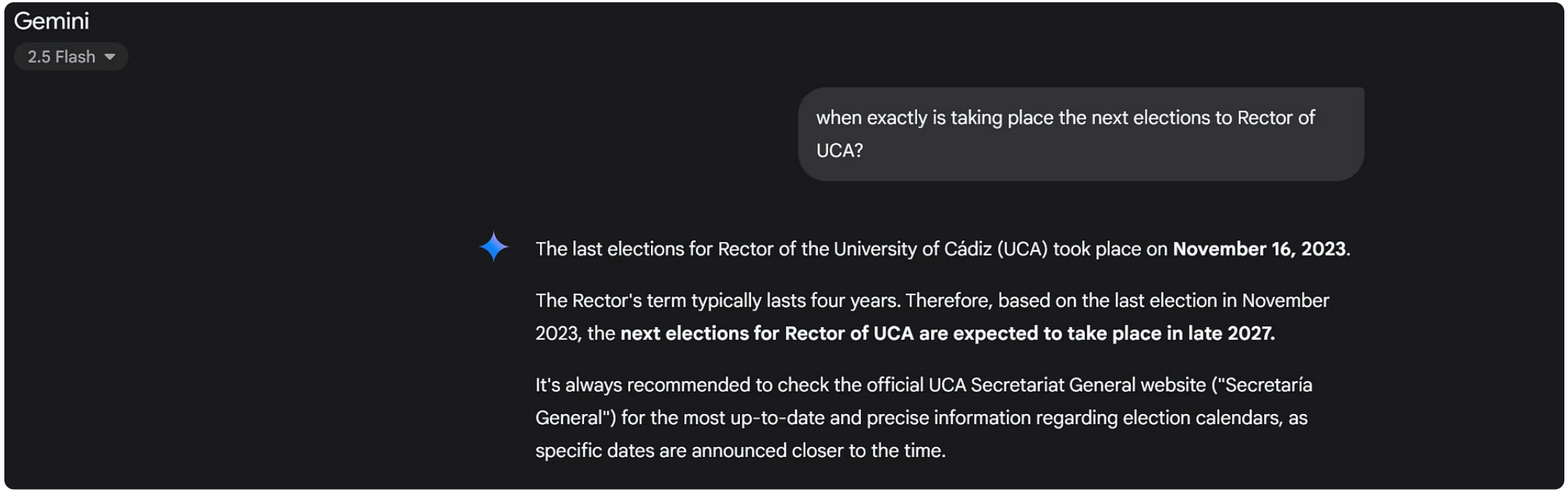
Retrieval Augmented Generation (RAG)

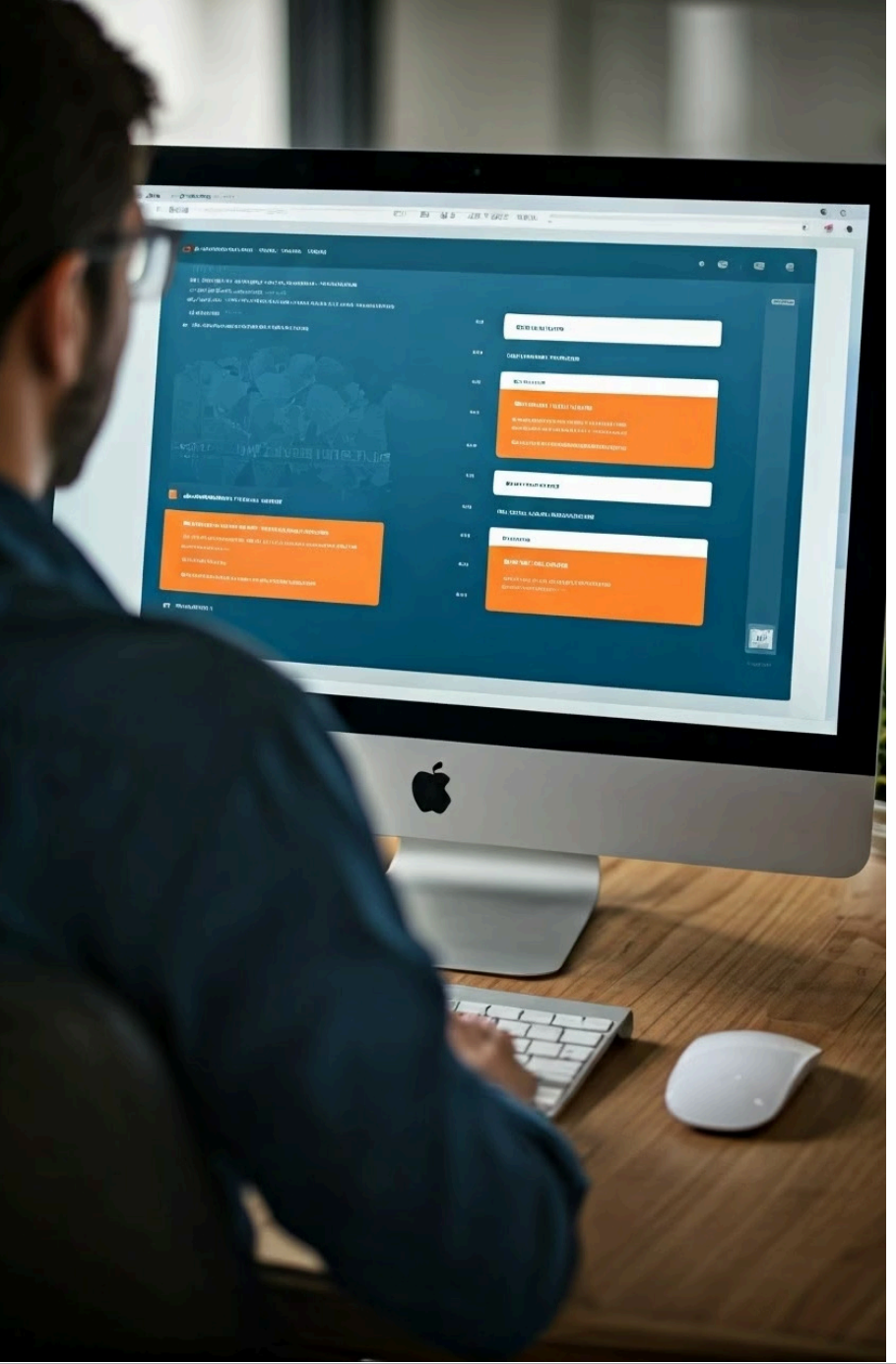
Iván Ruiz / Andrés Muñoz



Contents


Gemini
2.5 Flash
when exactly is taking place the next elections to Rector of UCA?
The last elections for Rector of the University of Cádiz (UCA) took place on **November 16, 2023**.
The Rector's term typically lasts four years. Therefore, based on the last election in November 2023, the **next elections for Rector of UCA are expected to take place in late 2027.**
It's always recommended to check the official UCA Secretariat General website ("Secretaría General") for the most up-to-date and precise information regarding election calendars, as specific dates are announced closer to the time.

Introduction



Information Quality: Issues

Outdated Information

LLMs have limited knowledge of the world. They cannot access information created after their training.

Inaccessible Information

Public web information that was not captured during the data extraction process prior to model generation is also inaccessible to LLMs.

Private Information

LLMs cannot access protected information from organisations, limiting their ability to provide complete responses.

Hallucinations

LLMs can generate incorrect, inaccurate or completely fabricated information that appears plausible.

Information Quality: Remediation

Fine-tuning

Fine-tuning involves **adjusting** the **parameters** of a pre-trained language model to adapt it to a specific dataset.

This method is ideal for improving the model's accuracy on a particular task, such as generating text with a **specific style or tone**.

Retrieval-Augmented Generation

A more **efficient** technique than fine-tuning to reduce hallucinations in LLMs.

It consists of augmenting the input given to the LLM, incorporating the entire knowledge base it needs to provide a coherent response.

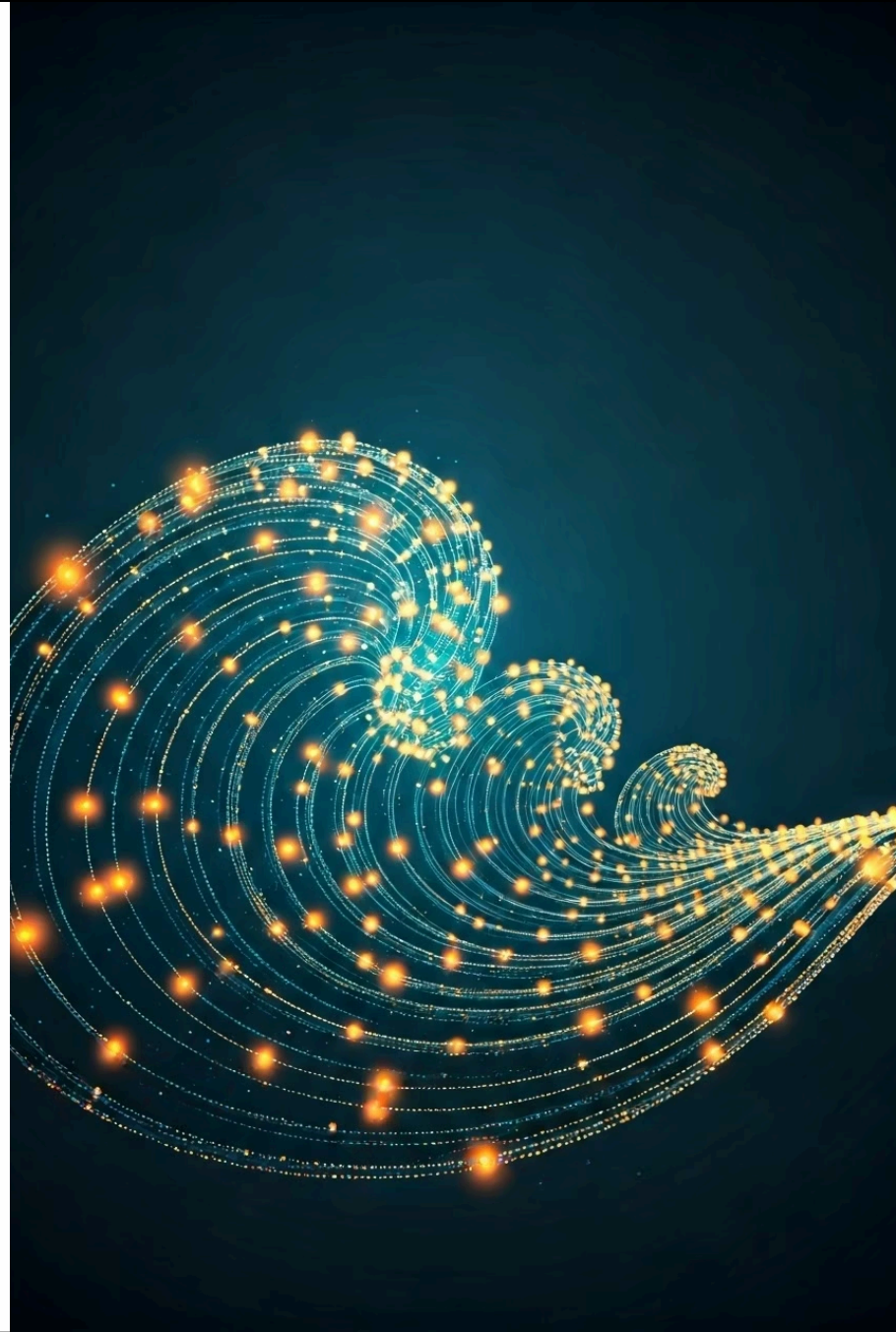
Retrieval-Augmented Generation (RAG)

Improved quality

Retrieval-augmented text generation seeks to improve the quality of language model responses.

Knowledge base

The language model connects to a knowledge base, retrieving relevant information to generate enriched responses.



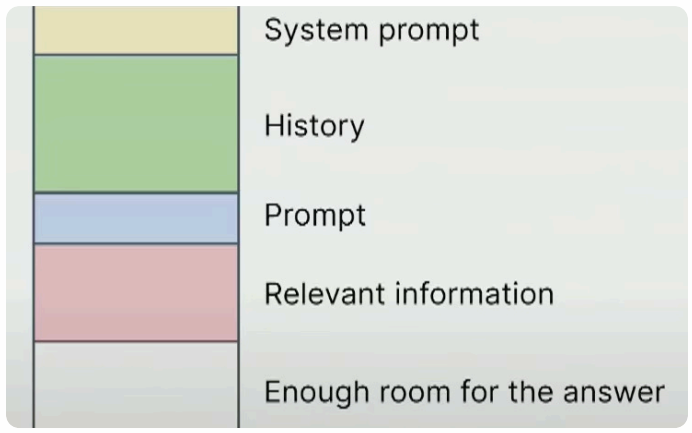
It's very easy to make RAG..

Ex 1. Creating an assistant with EasyRAG

...but getting good results is not so easy

Several problems that need to be addressed...

Challenges in Facing a RAG System



Context Window

We should not include full documents as the context window is not unlimited*.

Token Consumption

Long documents can consume many tokens, generating high costs and/or longer response times

Irrelevant Information

Including too much irrelevant information in the input prompt can reduce the accuracy of the responses provided by the LLM.

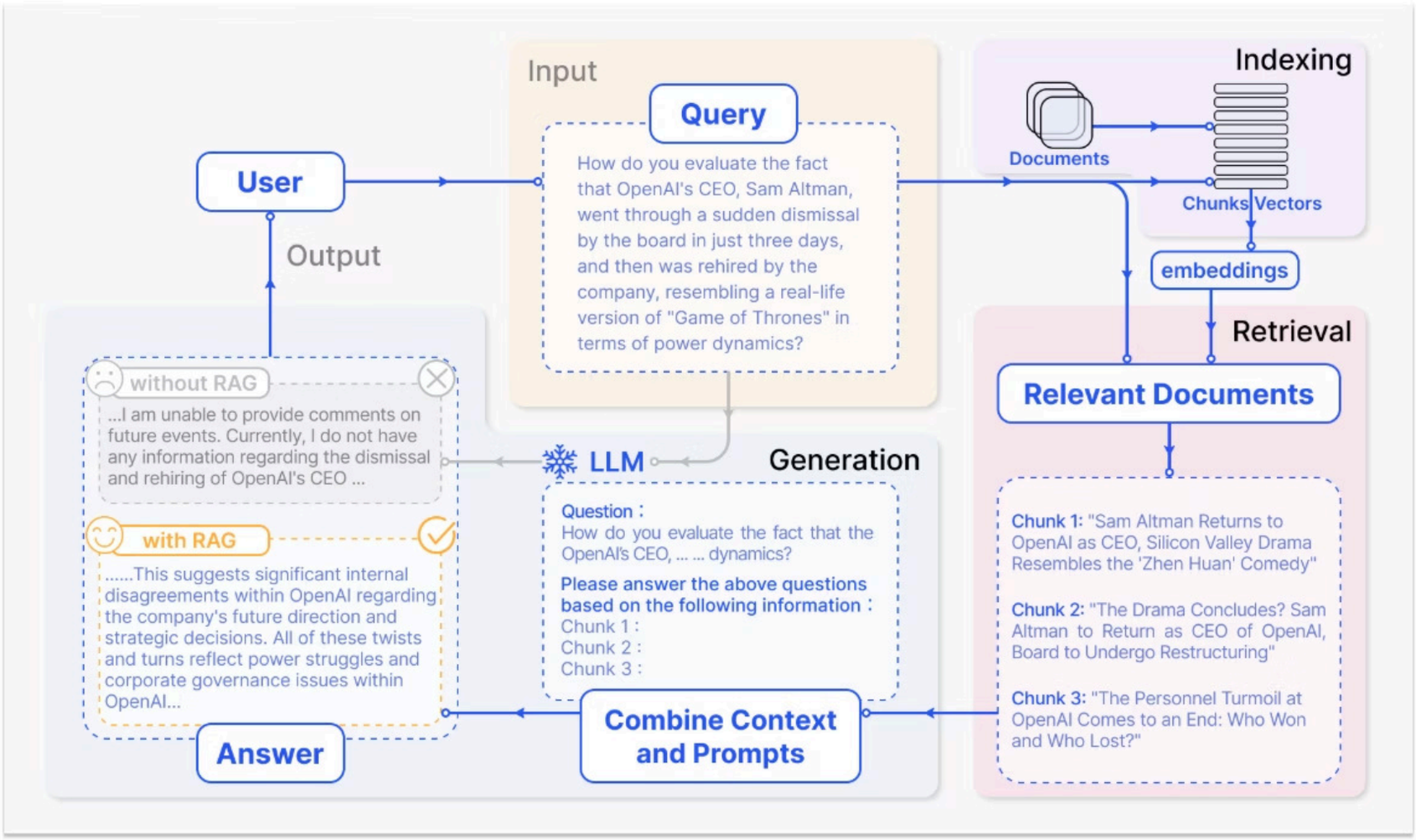
Question Complexity

The user might ask *"How do the advances made at the University of Cádiz in the field of biotechnology compare to those achieved by other universities, taking into account the publications in the last five years?"*

1


Naive RAG

Retrieval-Augmented Generation



Yunfan Gao et al. Retrieval-Augmented Generation for Large Language Models: A Survey, 2023

Stages of the RAG

Indexing

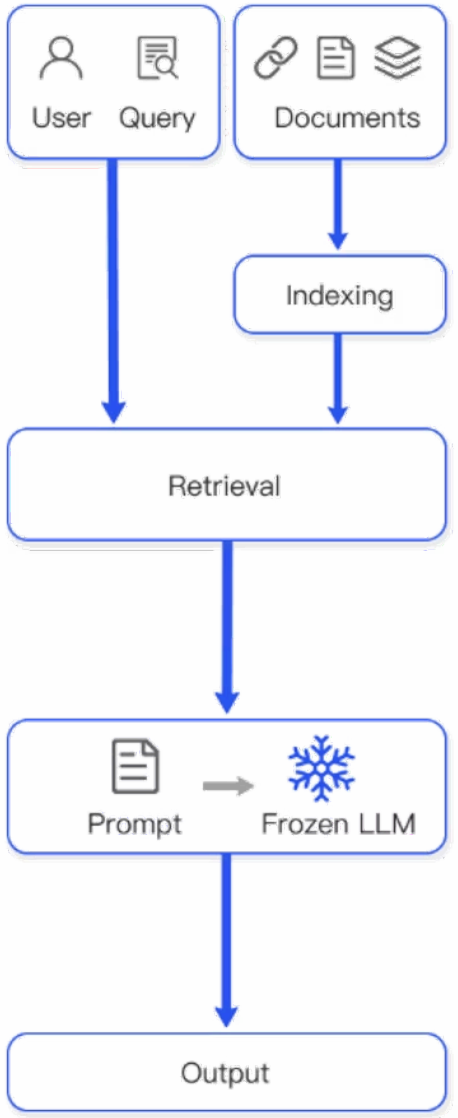
Offline pre-processing of information (documents) to enable the following stages

Retrieval

Retrieval of information that occurs online when the user launches the query to the LLM

Generation

Response synthesised by the LLM as a response to the query and the retrieved texts



Naive RAG

Information Indexing

Information Indexing

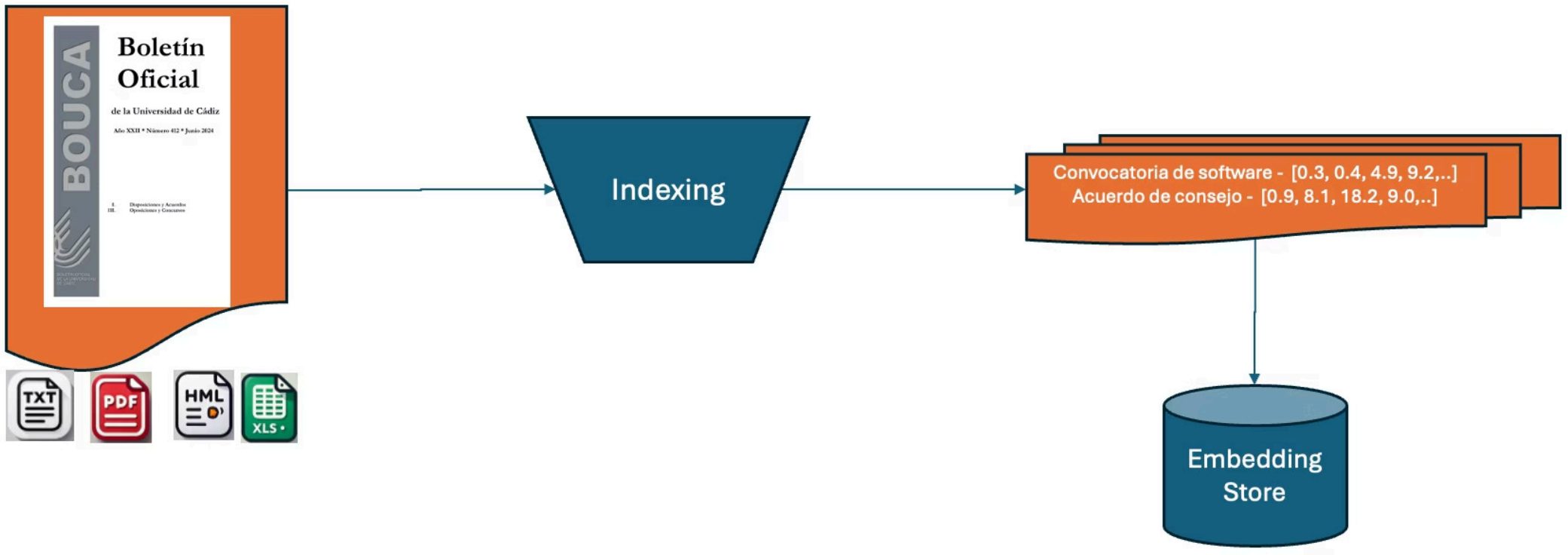
Transformation

Converting information into a format that enables efficient searches.

Importance

This process is fundamental for subsequent retrieval and text generation.

Indexing (example)

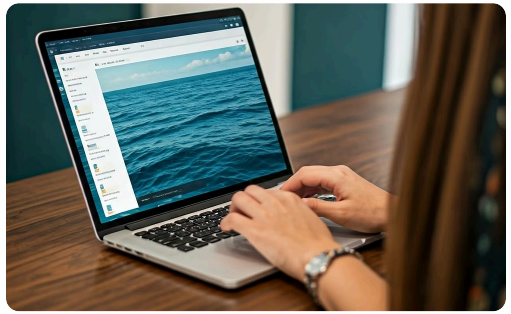


ETL

The indexing process can be divided into successive stages that are integrated into an ETL (Extract, Transform, Load) pipeline.

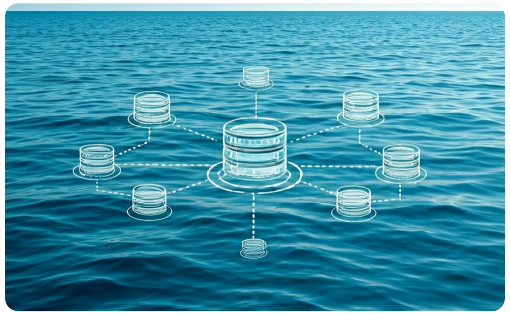
1. Documents are **loaded** from external sources.
2. Documents are **analysed** using some parser.
3. Documents are **transformed** to clean or enrich them with metadata.
4. Documents are **divided** into smaller segments.
5. Numerical vectors (**embeddings**) are generated for each segment.
6. The vectors are **stored** in a dedicated database.

Loading



File systems

Information collected in files stored in local file systems or remote systems (e.g. NextCloud or GDrive)

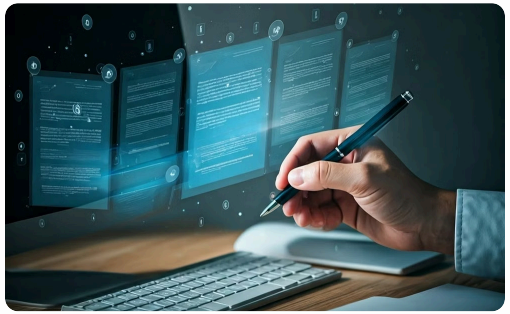


Database systems

Information collected in tables in relational databases (e.g. Oracle) or in other formats of non-relational databases (e.g. MongoDB documents)

Storage services

Information collected in open source object storage systems (e.g. MinIO) or proprietary (e.g. AWS S3 or Google Cloud Storage)



Document managers

Open source document management systems (e.g. Alfresco) or proprietary (e.g. SharePoint)

Parsing

Documents must be analysed and their data extracted with the help of an appropriate parser.

Unstructured information

Files with layout and images (e.g. DOC, PPT or PDF)

Semi-structured information

Files with some form of tagging (e.g. JSON, XML or HTML)

Structured information

Databases accessible with SQL, spreadsheets or CSV files

Transforming: Cleaning

Removing irrelevant information

Irrelevant headers or footers, watermarks or digital signatures should be removed to optimise the content.

Extracting visual elements

Tables, images or graphs should be properly processed to generate text.

Normalising the text

Extra spaces, strange characters are removed and the format is normalised for coherent analysis.

Removing HTML markup

To have the text ready for indexing, HTML tags are removed.

Transforming: enriching

Enriching with metadata is essential for:

- Providing additional context to the LLM.
- Optimising searches with filtering.
- Re-indexing information after updates to the source documents.

Common Metadata

URL

Access the source of the document and verify its content.

File name and page number

Uniquely identify the document and/or access a specific section.

Retrieval date

Determine the timeliness of the document.

Category

Facilitate the classification and search of the document by the LLM.

Authorship

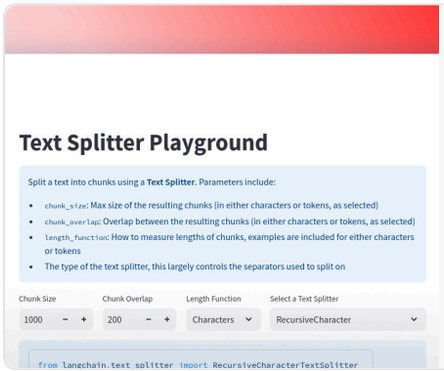
Evaluate the reliability of the content.

Identifiers at source

Efficiently locate information records in the source sources

Chunking (splitting)

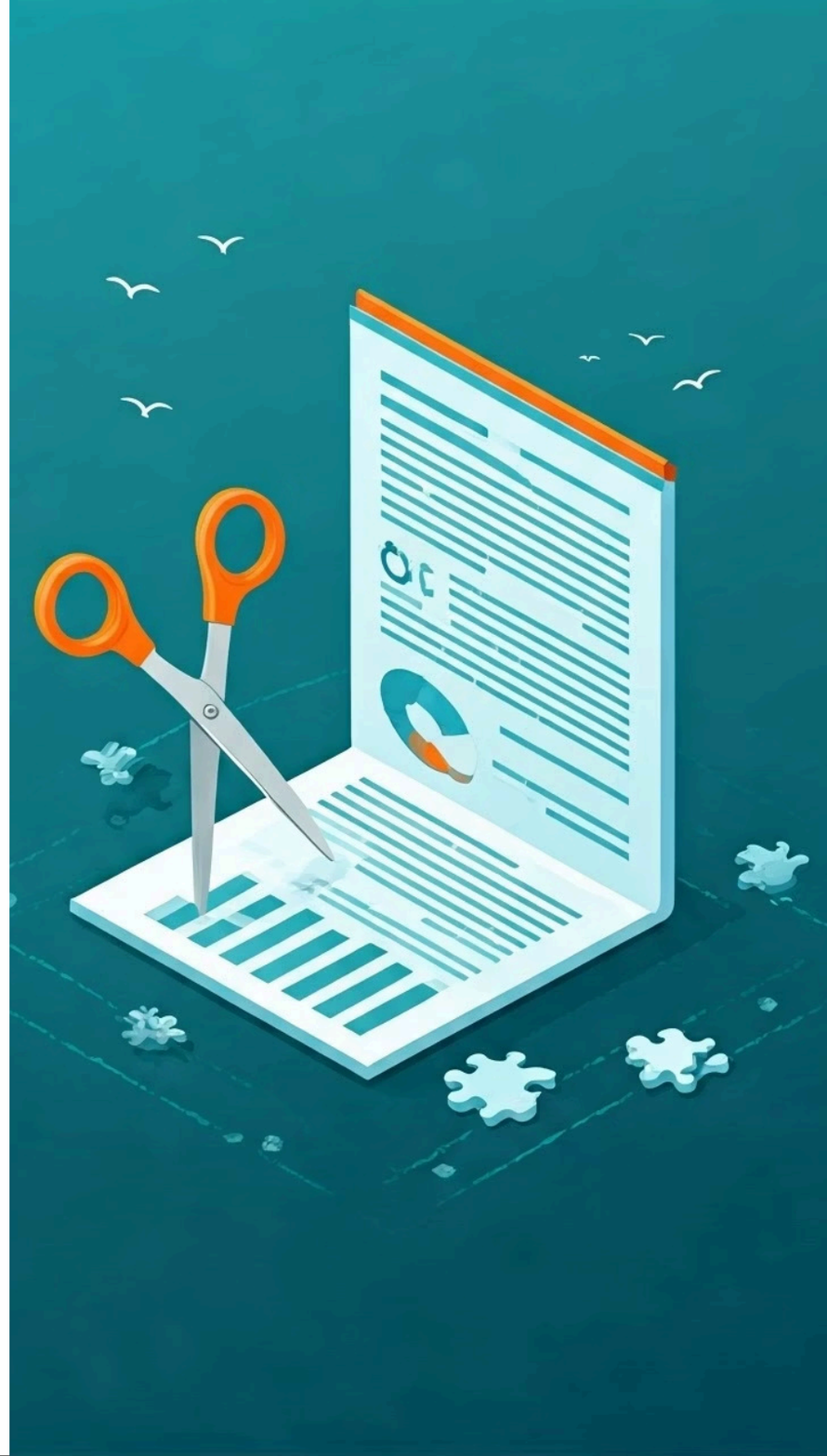
- **Fixed size:** division into fixed-length segments of characters or tokens (approximately 300).
- **Recursive:** division into paragraphs, lines, spaces and, finally, into words.
- **Document-based:** division by structural delimiters of the document. Examples:
  - **#** in Markdown
  - **<section>** in HTML
  - **\section** in LATEX
  - **class** in Python



Streamlit

splitter

ui.png



Indexing: embedding creation

Semantic Representation

Embeddings capture the meaning of text, representing it as numerical vectors of a certain length (1536 floats).

Embedding Models

Specific machine learning models are used to generate the embeddings.

Google's Word2Vec

Google's Word2Vec was one of the first embedding models.

Types of Tasks

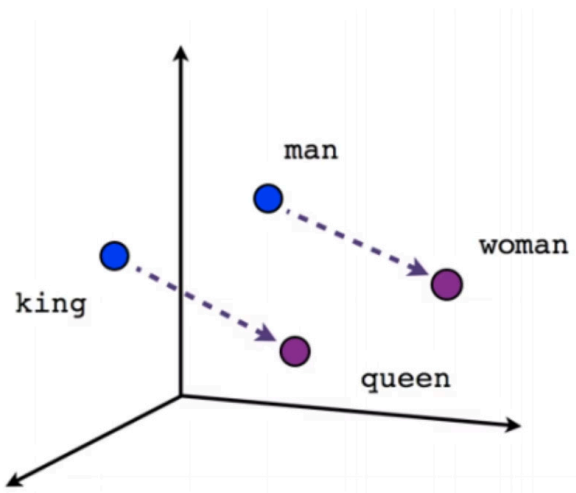
In addition to information retrieval, embeddings can be used to perform classification or clustering tasks.

Local Execution

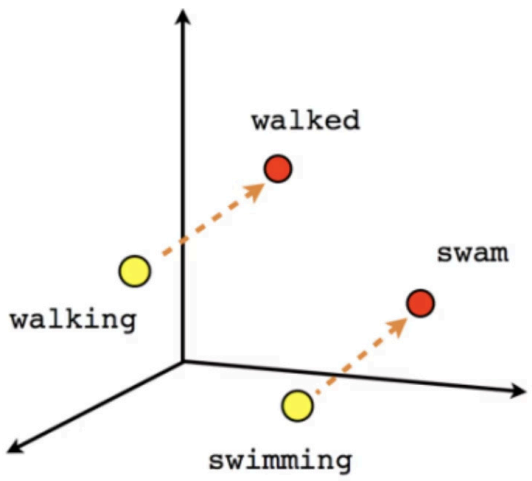
There are embedding models that can be executed locally (ONNX Runtime or Jllama) in the same execution process.

Remote Execution

We can use HTTP APIs to access embedding models, such as OpenAI's *text-embedding-3-small*.



Male-Female



Verb tense

Ranking of embedding models

Choosing the most appropriate embedding model is also an important decision...

huggingface

MTEB Leaderboard – a Hugging Face Space by mteb

Discover amazing ML apps made by the community

Indexing: storing embeddings

Storage

The generated embeddings must be stored along with the corresponding text segments to enable information retrieval.

Vector stores

Vector stores are specialised databases for storing high-dimensional vectors, such as embeddings.

Similarity search

These systems allow for searches based on the similarity between vectors.

Integration

Vector stores offer APIs for integration with different programming languages, which facilitates their use in AI applications.

Comparison of Vector Databases

And choosing the right database...

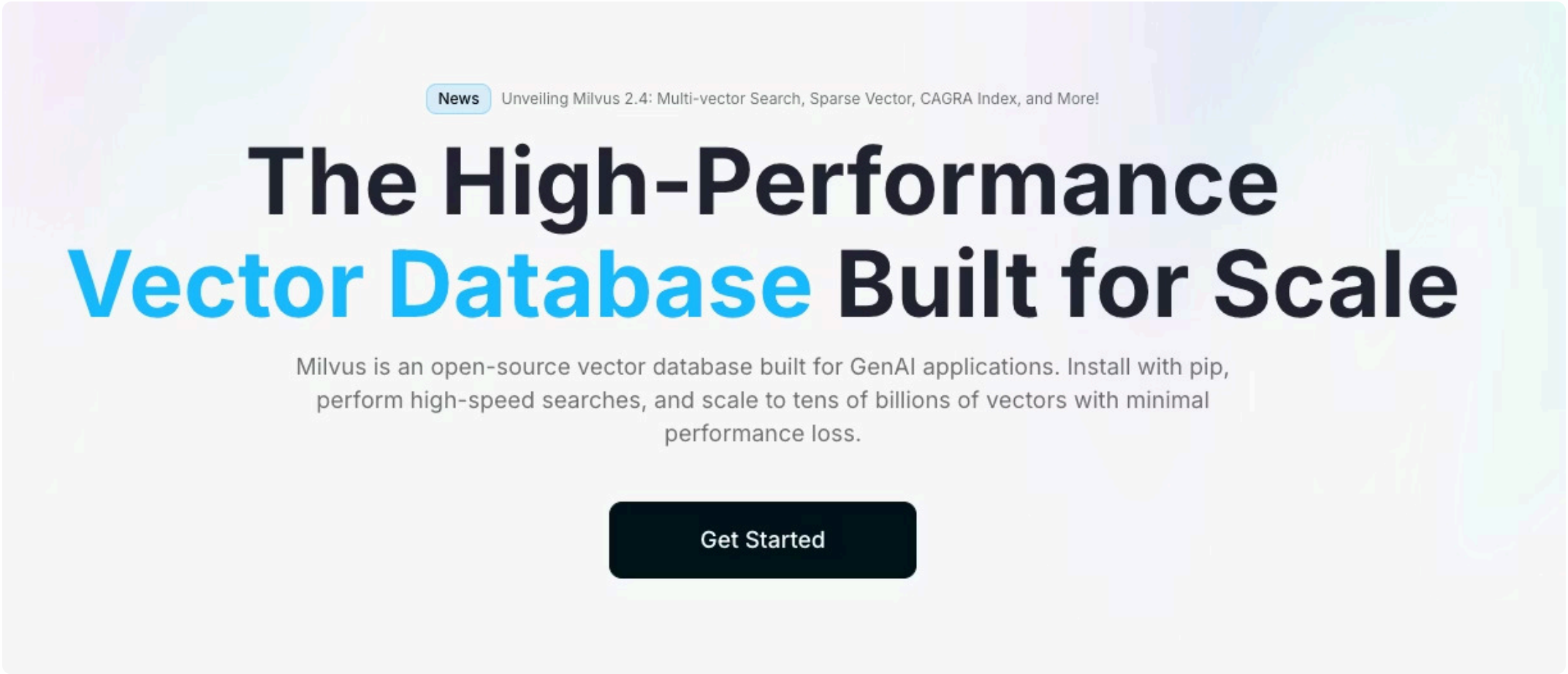
superlinked.com

Vector DB Comparison

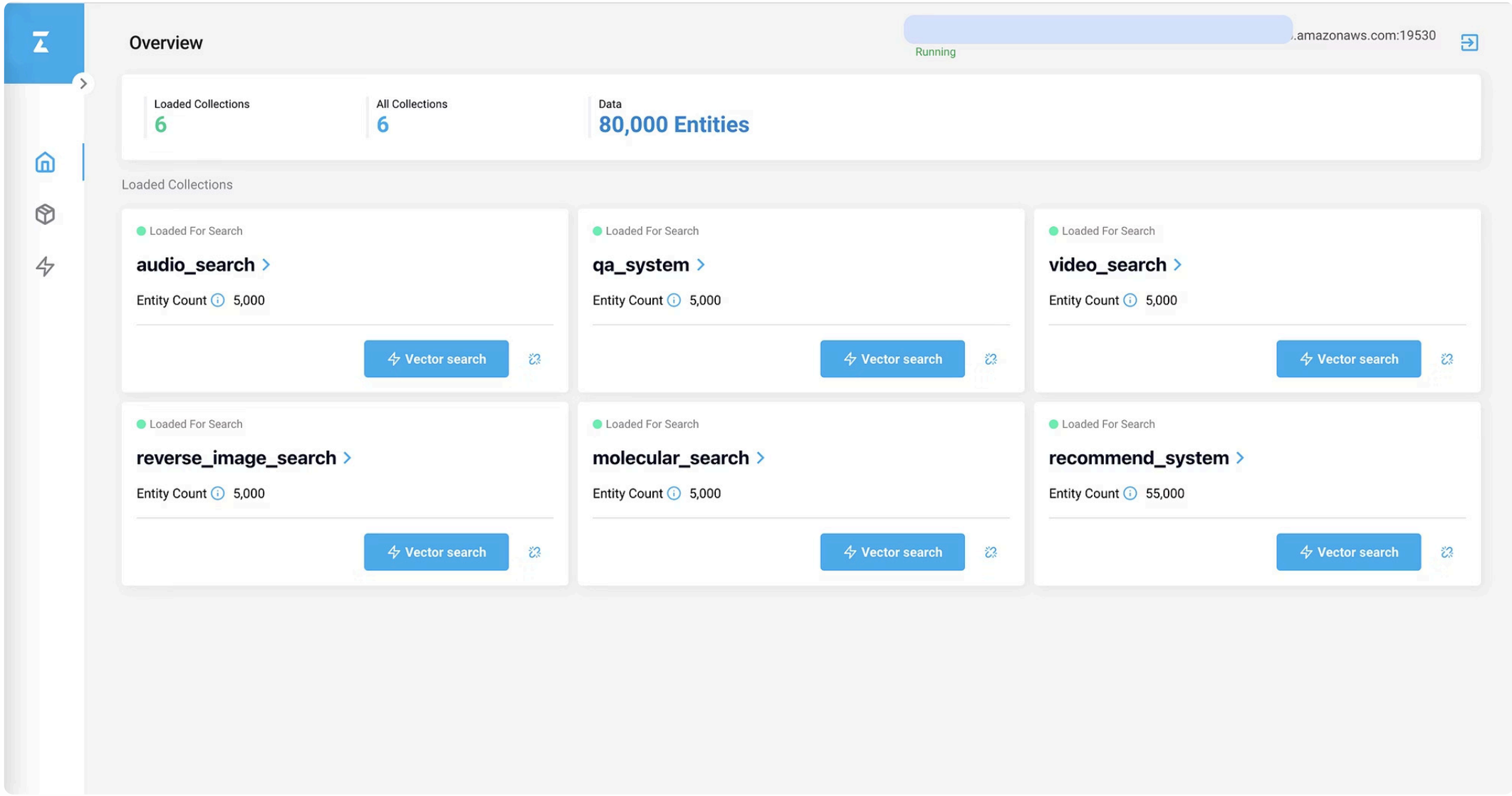
Vector DB Comparison is a free and open source tool from VectorHub to compare vector databases.

Ex 2. Use of an embedding store

Milvus Vector Database



Attu: Milvus Graphic Client



Installing Milvus and Attu

Download the Docker script

```
curl -sfL https://raw.githubusercontent.com/milvus-io/milvus/master/scripts/standalone_embed.sh -o standalone_embed.sh
```

Add the user.yaml file

```
# Extra config to override default milvus.yaml
common:
  security:
    authorizationEnabled: true
```

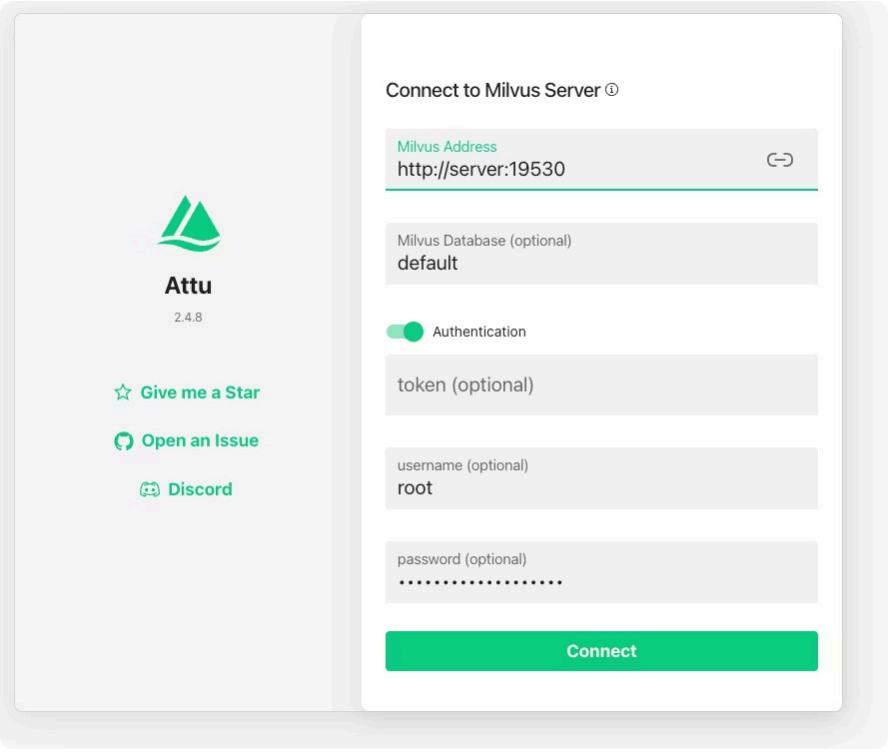
Start the Milvus server

```
bash standalone_embed.sh start
```

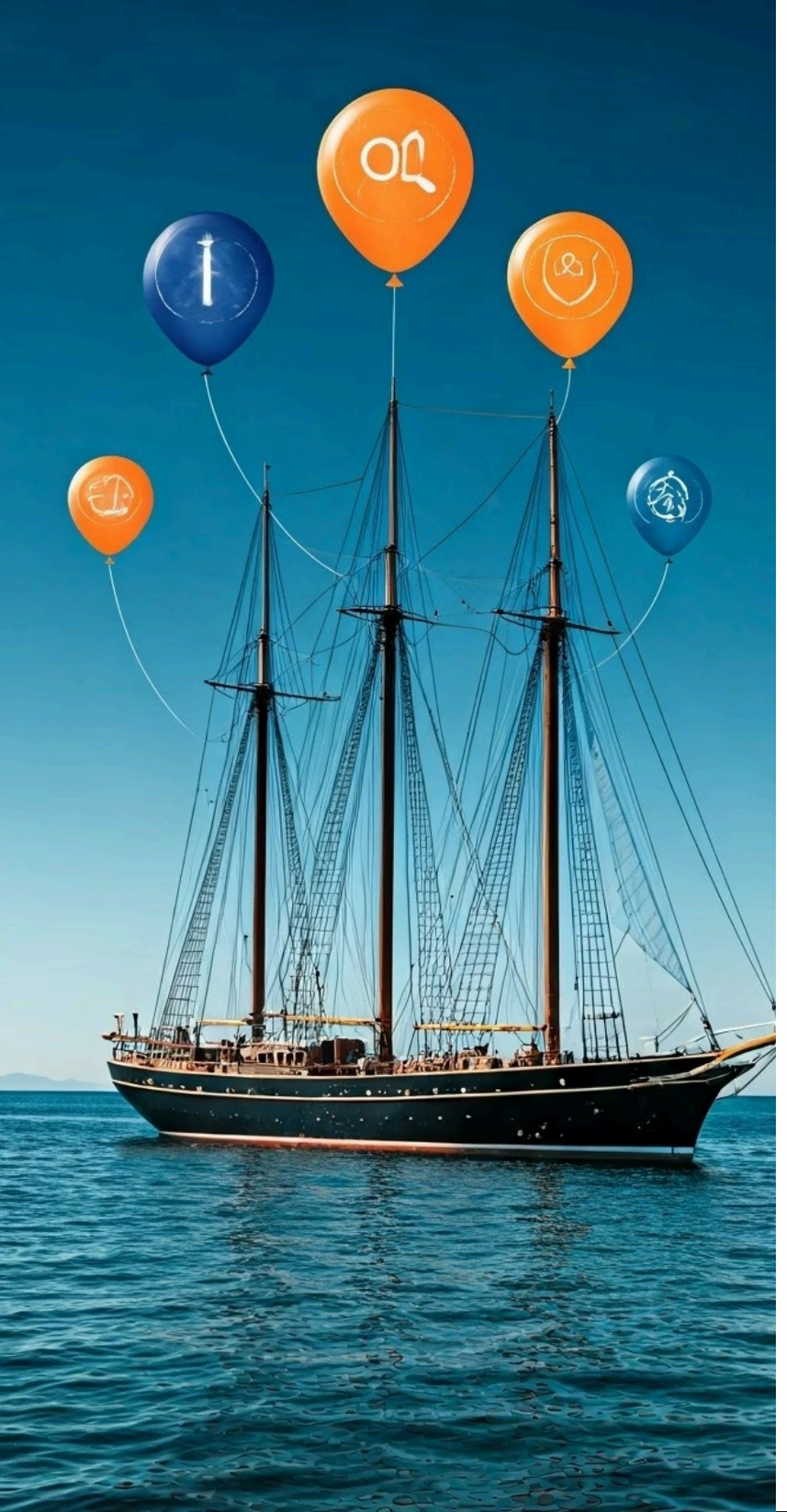
Download the Attu image and run it

```
docker run -d -p 8000:3000 -e MILVUS_URL=http://SERVER:19530 zilliz/attu:v2.4.8
```

Open a browser and access http://SERVER:8000 with the root user and Milvus password



Information Retrieval



Data Sources

Vector Databases

Perform semantic searches with filters.

Web

Search engines like Google, Bing or DuckDuckGo.

Relational Databases

Access data collected in engines (Oracle, PostgreSQL) through SQL queries.

NoSQL Databases

Queries in NoSQL databases, such as Neo4J, using Cypher language.

Search Servers

Search engines with storage capabilities, such as Solr or ElasticSearch.

Web Services

HTTP APIs to obtain up-to-date information.

Information Retrieval

Information retrieval is a fundamental part of the QA process, as it allows access to relevant data for generating information.

The quality of the retrieved information directly affects the accuracy and quality of the generated text.

SQL Search

- With the SQL language, you can perform searches in string fields using operators like LIKE or REGEXP_LIKE
- Exact or partial matches of keywords within the documents are searched, according to the defined patterns.

```
SELECT * FROM products WHERE description LIKE '%camera%';
```

```
SELECT * FROM products WHERE REGEXP_LIKE(description, 'camera|phone');
```

```
SELECT * FROM products WHERE CONTAINS(description, 'camera') > 0;
```

Full-text search

Full-text searches look for the occurrence of keywords in documents.

These systems assign a score to documents based on the frequency and relevance (TF-IDF algorithm) of the keywords in the query.

The score is based on factors such as frequency of appearance, creation date, proximity of keywords to each other, geolocation, etc.

They have limitations when working with natural language queries due to the difficulty of identifying synonyms, paraphrases or context.

elasticsearch

Semantic search in vector stores

Embeddings

Documents are converted into numerical vectors that represent their meaning.

Vector Stores

Allow for fast and efficient searches in the vector space, based on the similarity of vectors.

Similarity Distance

Documents are ranked based on the similarity of their embeddings to the embedding of the input query, using angular distance (cosine similarity).

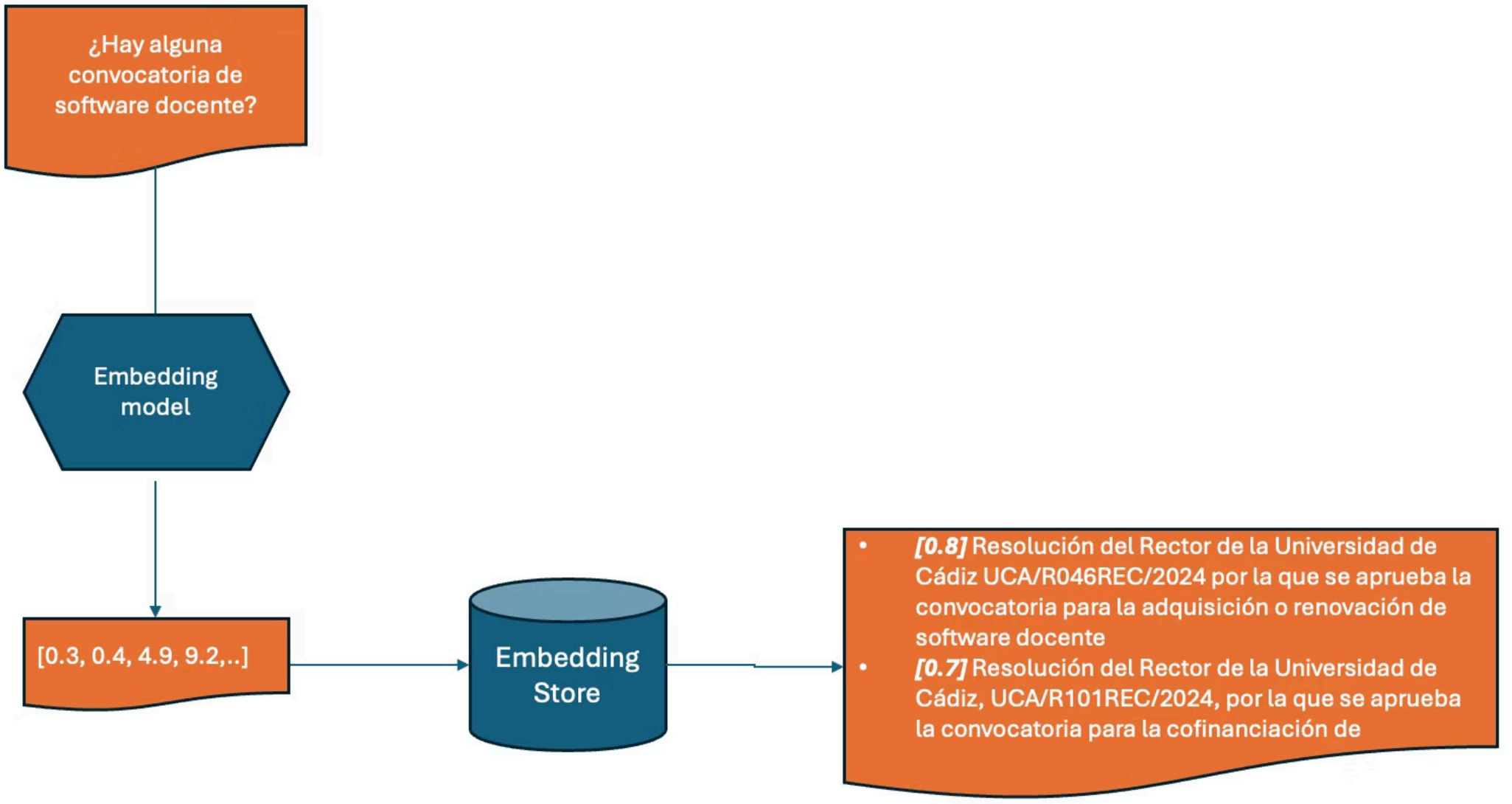
Different embedding models produce different outputs

Do not compare embeddings generated with a particular model to embeddings generated by others.

Obtaining Results

The k most similar chunks to the query are obtained, providing a set of relevant information.

Retrieval (example)



Ex 3. Creation of a semantic search engine based on a vector database

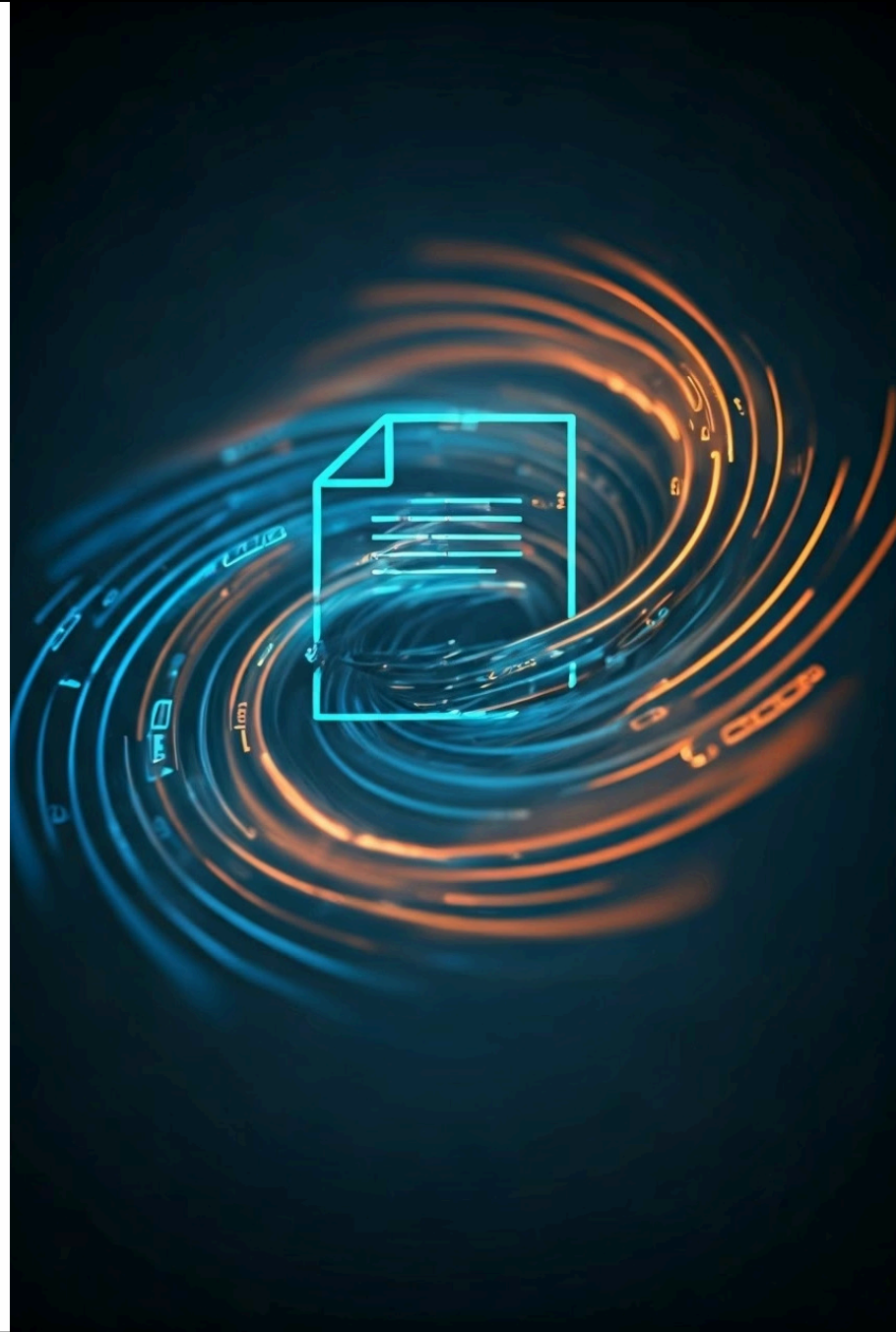
Semantic searches combined with filters

User Query

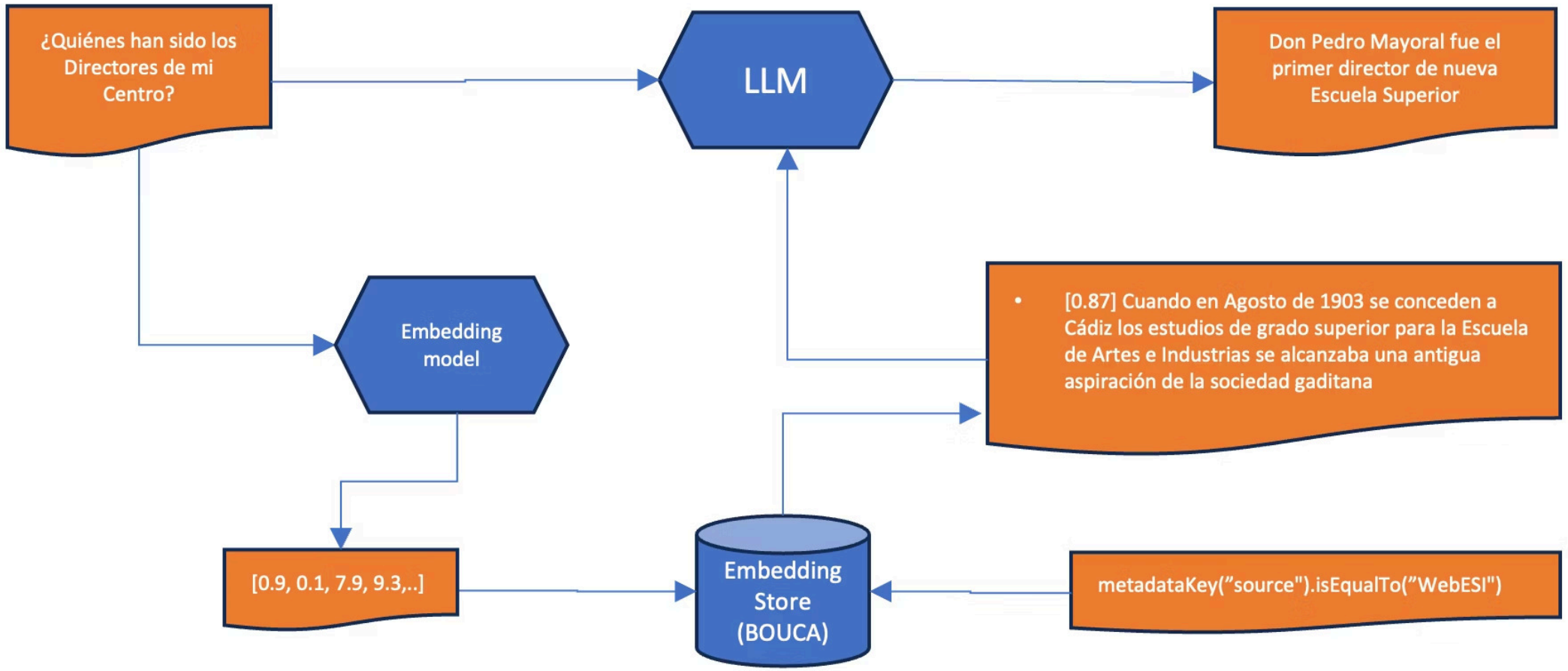
The metadata accompanying the text segments and their embeddings enable the possibility of applying additional filters

Metadata Filters

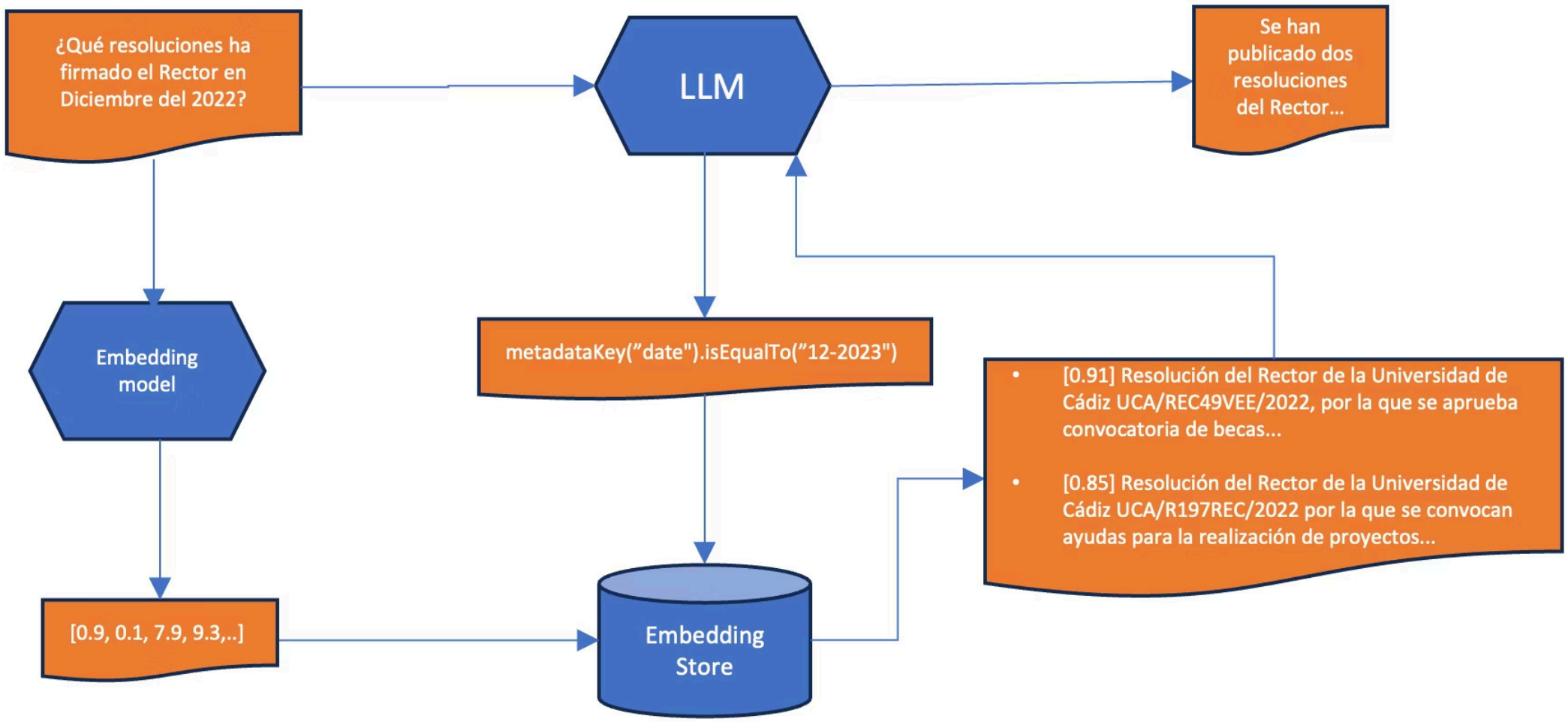
Filters can be defined statically or dynamically (e.g. according to user permissions) or generated by an LLM



Metadata Filters

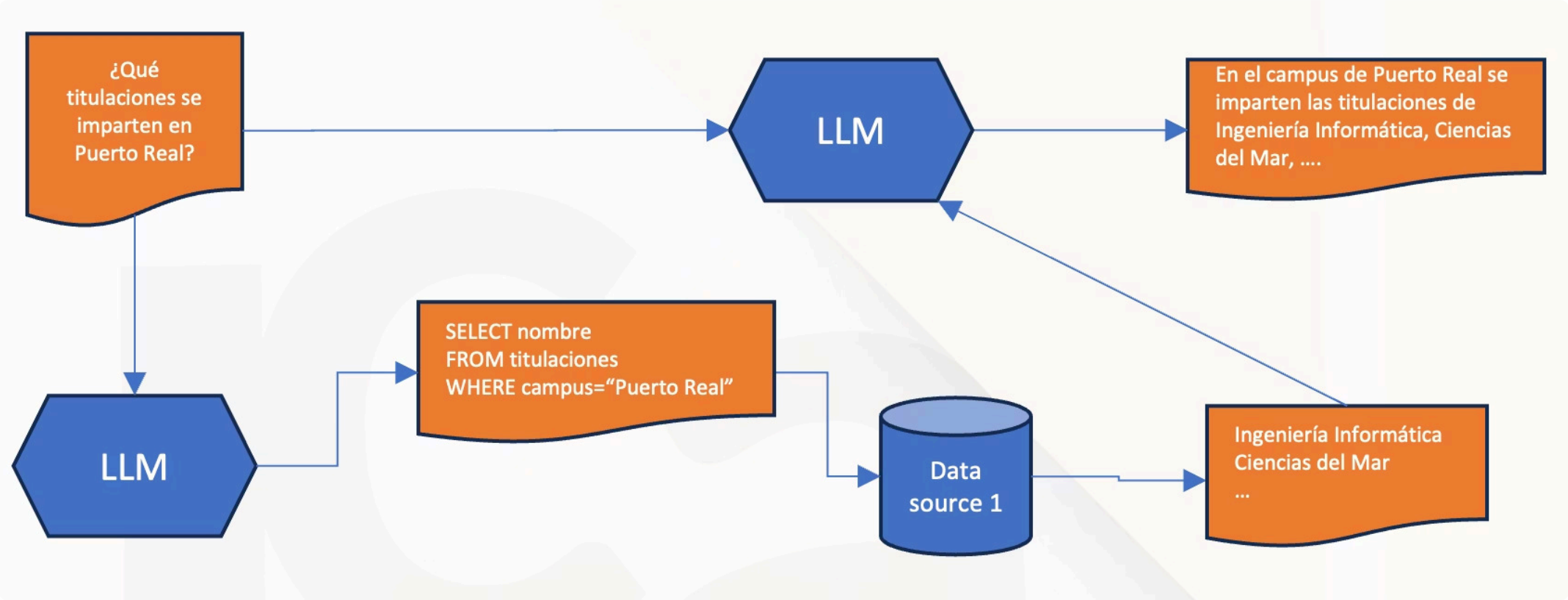


Generated Filters (Self-Querying)



Semantic Search in Relational Databases

The user makes a natural language query that an LLM translates into a valid SQL statement.

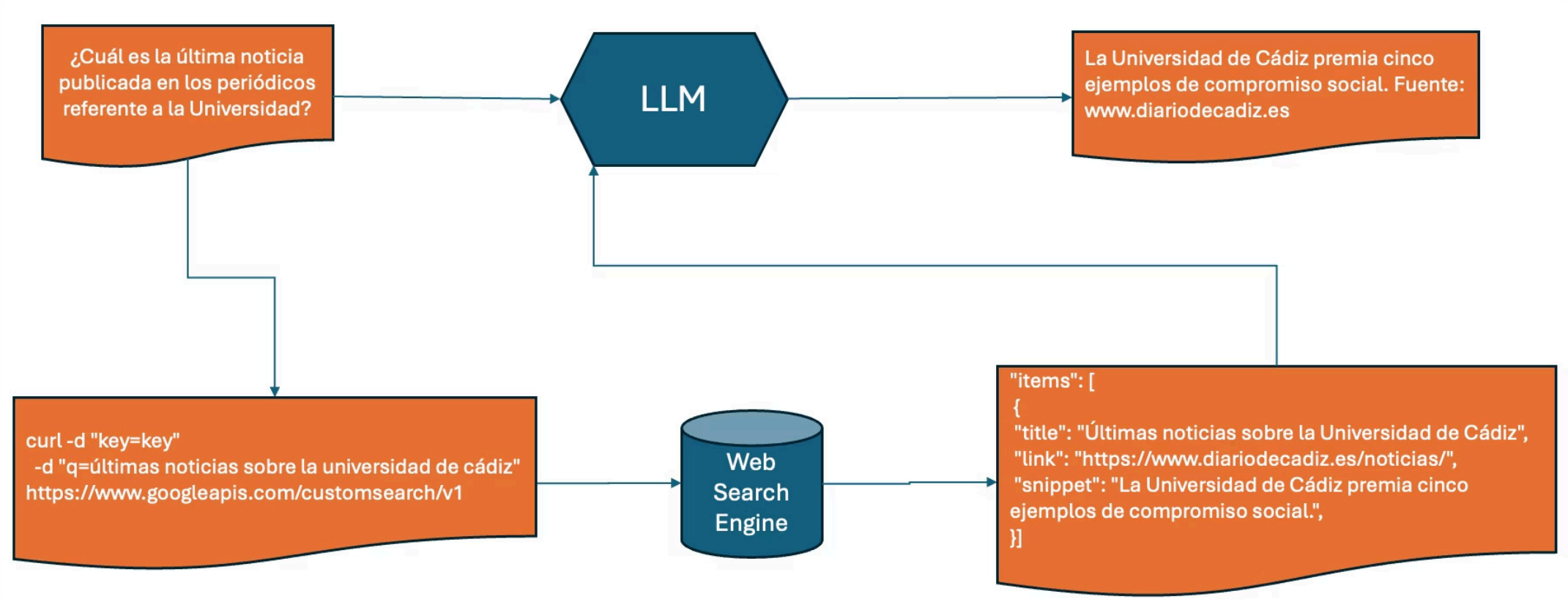


Semantic Search in Relational Databases: system prompt

"You are an expert in writing SQL queries.
You have access to a {{sqlDialect}} database with the following structure:
{{databaseStructure}}
If a user asks a question that can be answered by querying this database, generate an SQL SELECT query. Do not output anything else aside from a valid SQL statement!"

Ex 4. Creating a semantic search engine in a relational database

Semantic search on the web



Web search platforms

Search engines

Google, Bing, and DuckDuckGo offer APIs to access online information.

Web search APIs

Tavily, SerpApi, Serper and SearchApi are platforms specialised in web information search.

Web search APIs

These APIs allow applications to integrate search functions without developing their own engine.

Information Generation

Information Generation

Information generation is a complex process that requires understanding natural language, processing information, and generating creative text, adapting the language and style to the user's needs.

The LLM combines the input prompt with relevant information to generate the output.

The quality of the output depends on the quality of the information retrieved and the capability of the LLM itself.

Information Generation

The LLM must be instructed with a specific prompt in order to generate the desired response

"

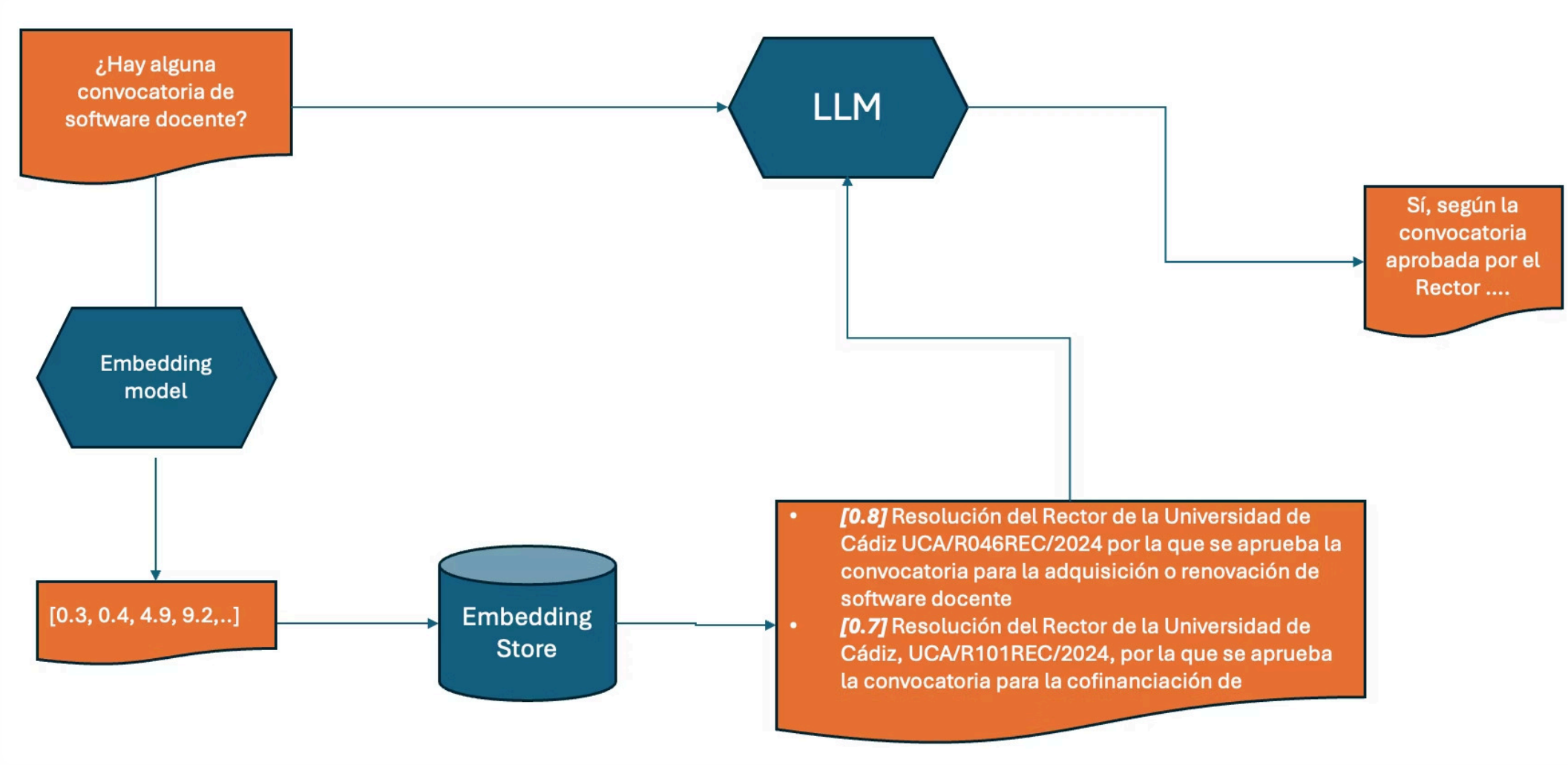
...
Answer the question based only on the following context:

{context}

Question: {question}

"

Generation (example)



Ex 6. Creating a RAG assistant


Advanced
RAG

Limitations of the Basic RAG

1 **Irrelevance**

Retrieved texts may be of little relevance or lack vital information.

2 **Redundancy**

Similar information from different sources generates unnecessary duplication.

3 **Stylistic Inconsistency**

Variations in style and tone make it difficult to achieve coherence in the final result.

4 **Information Overload**

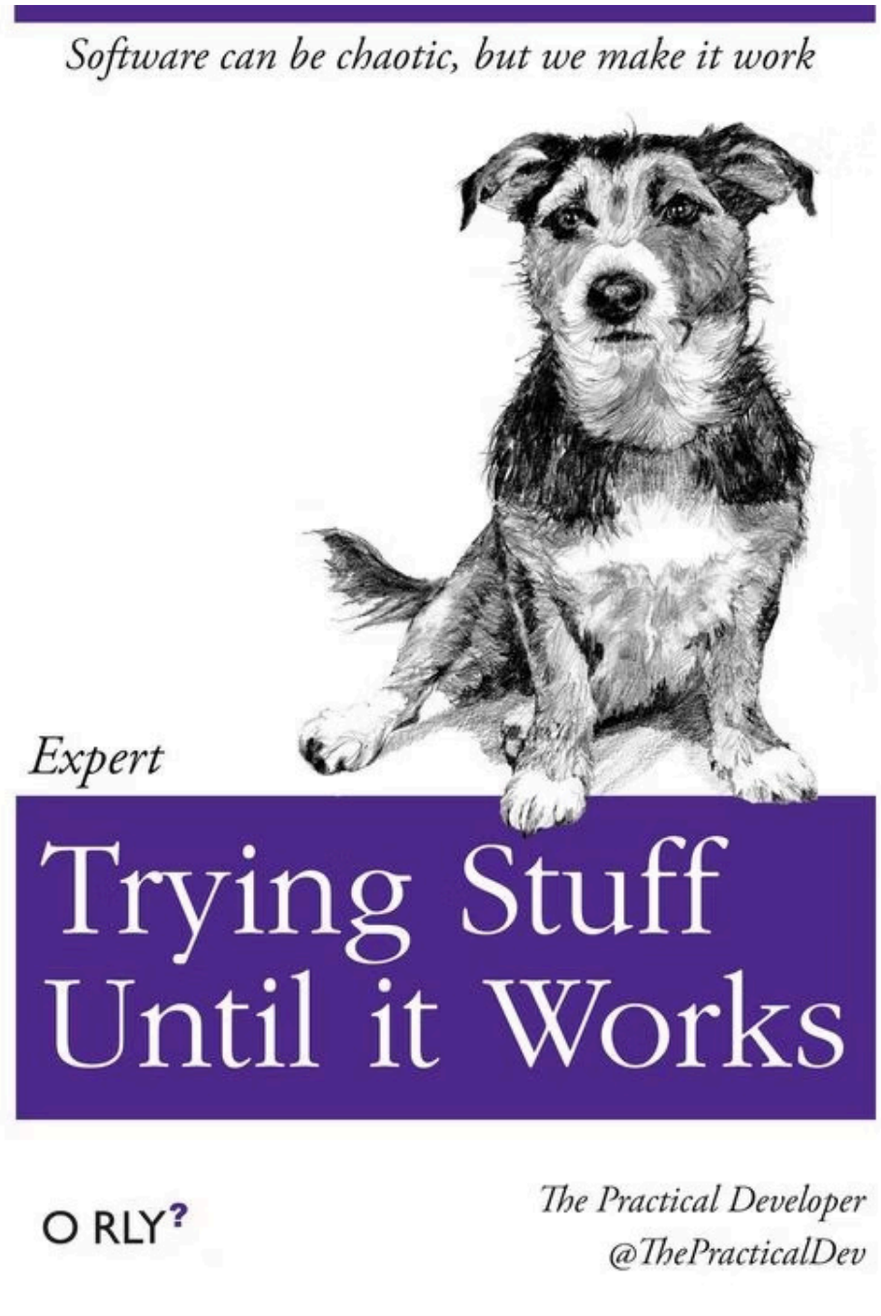
Excess data can dilute the importance of key information.

5 **Unanswerable Questions**

Complex queries with multiple parts cannot be resolved.

6 **Implicit Questions**

Queries may refer to implicit and unknown information for the retriever.


Software can be chaotic, but we make it work
Expert
Trying Stuff Until it Works
O RLY?
The Practical Developer
@ThePracticalDev

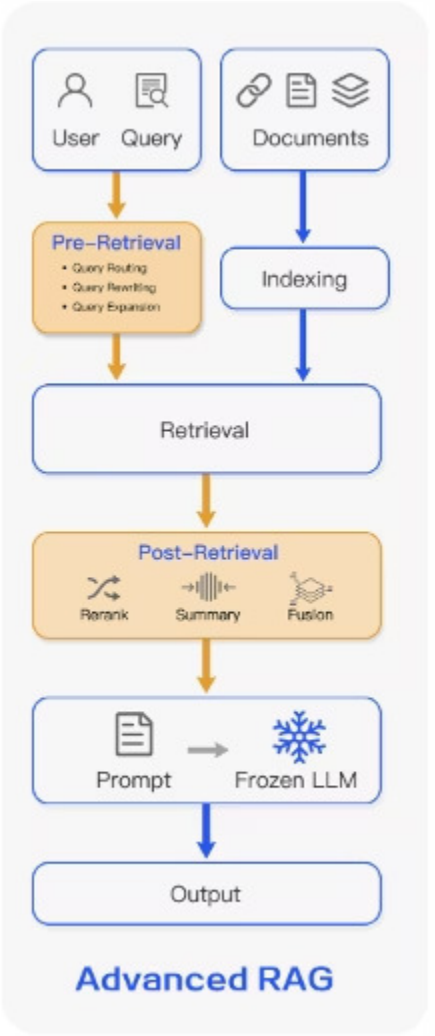
Advanced RAG Techniques

Pre-retrieval

Techniques applied before information retrieval.

Post-retrieval

Methods used after retrieval.



Pre-Retrieval Techniques

Pre-Retrieval Techniques

Query Rewriting

Query reformulation is a crucial technique for optimising search and information retrieval.

Query Expansion

Expanding the search involves adding additional terms to the original query to improve the precision and comprehensiveness of the results.

Query Compression

Query compression involves contextualising the original query to optimise the efficiency and precision of the search.

Query Routing

Routing the query to the most relevant data sources, optimising efficiency and relevance.

Query Rewriting: Query Reformulation

1

Identification

The original user query is analysed to detect areas for improvement.

2

Processing

An LLM modifies the query by including related terms and recognised entities.

3

Optimisation

A paraphrase is performed to align with the vocabulary of the knowledge base.

4

Refinement

Spelling and grammatical corrections are applied to improve accuracy.

Query Rewriting: Example

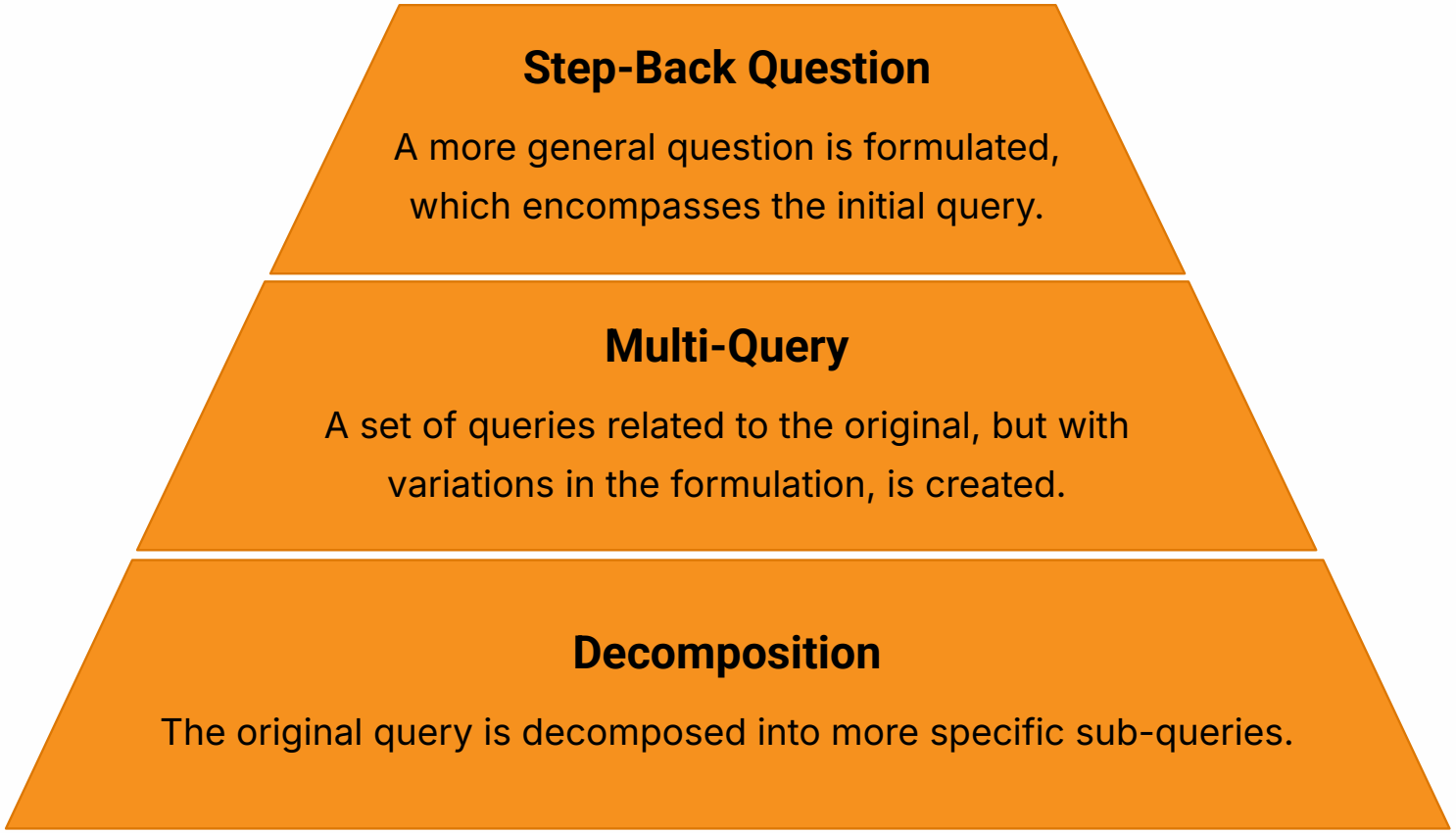
Original Query

"Renewable projects Cádiz university, research groups?"

Modified Query

"What are the renewable energy research projects being developed by the research groups at the University of Cádiz?"

Query Expansion: Broadening the Search



Step-Back Question

1

Specific Query

It starts from the original user's question.

2

Generalisation

A more abstract and broader question is formulated.

3

Dual Retrieval

Results are obtained from both the original question and the generalised one.

4

Synthesis

The information is combined to provide a more complete and contextualised answer.

Step-Back Question: Example

Original Query

"What specific strategies can a first-year engineering student apply to improve their performance on mathematics exams at the University of Cádiz?"

Additional Query

"What are the factors that influence the academic performance of first-year students at the University?"

Multi-Query

1

Initial Query

It starts from the original user's question.

2

Generation

Multiple variants are created while maintaining the level of abstraction.

3

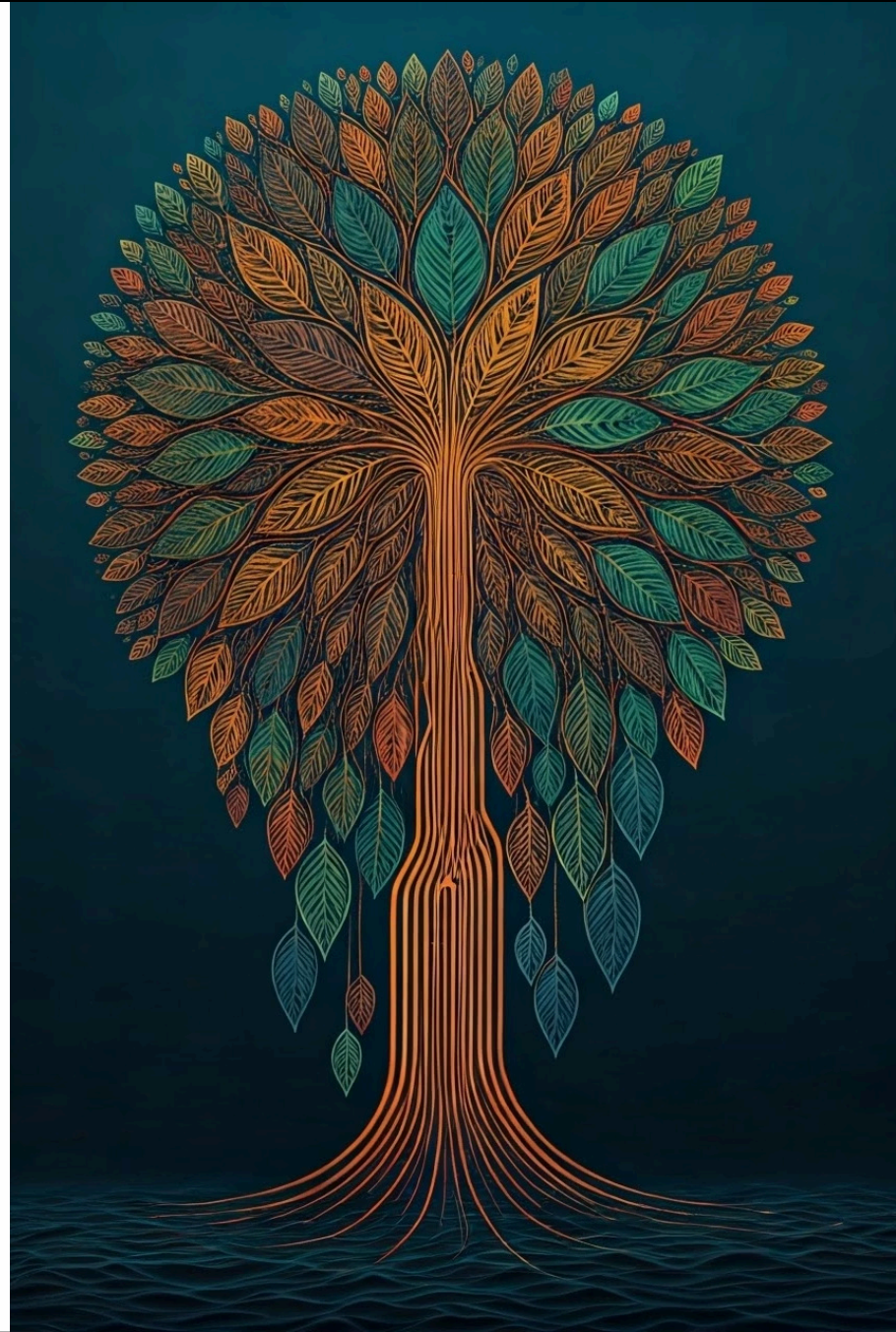
Diversification

Different perspectives of the same query are addressed.

4

Retrieval

All variants are used to obtain more complete results.



Multi-Query: Example

Original Query

"What exchange programmes does the University of Cadiz offer?"

Generated Queries

"What international agreements does the University of Cadiz have with European universities?"

"What requirements are needed to participate in the Erasmus+ programme?"

"Are there exchange opportunities outside of Europe, such as in America or Asia?"

"What financial support or scholarships does the University of Cadiz offer for exchange students?"

Query Decomposition

1

Analysis

The original query is examined to identify key components.

2

Decomposition

It is divided into more specific and concrete sub-questions.

3

Processing

Parallel or sequential queries are performed depending on the nature of the sub-questions.

4

Retrieval

All sub-questions are used to obtain more comprehensive results.

Query Decomposition: example

Original Query

"What are the most common evaluation methods in science degrees at the University of Cádiz and how do they compare to the methods used in humanities degrees?"

Derived Sub-queries

"What are the most common evaluation methods in science degrees at the University of Cádiz?"

"What are the evaluation methods used in humanities degrees at the University of Cádiz?"

"How do the evaluation methods compare between science and humanities degrees?"

Query Compression: Contextualising the Query

1. **Context Analysis**

   The previous conversation history is examined.

2. **Compression**

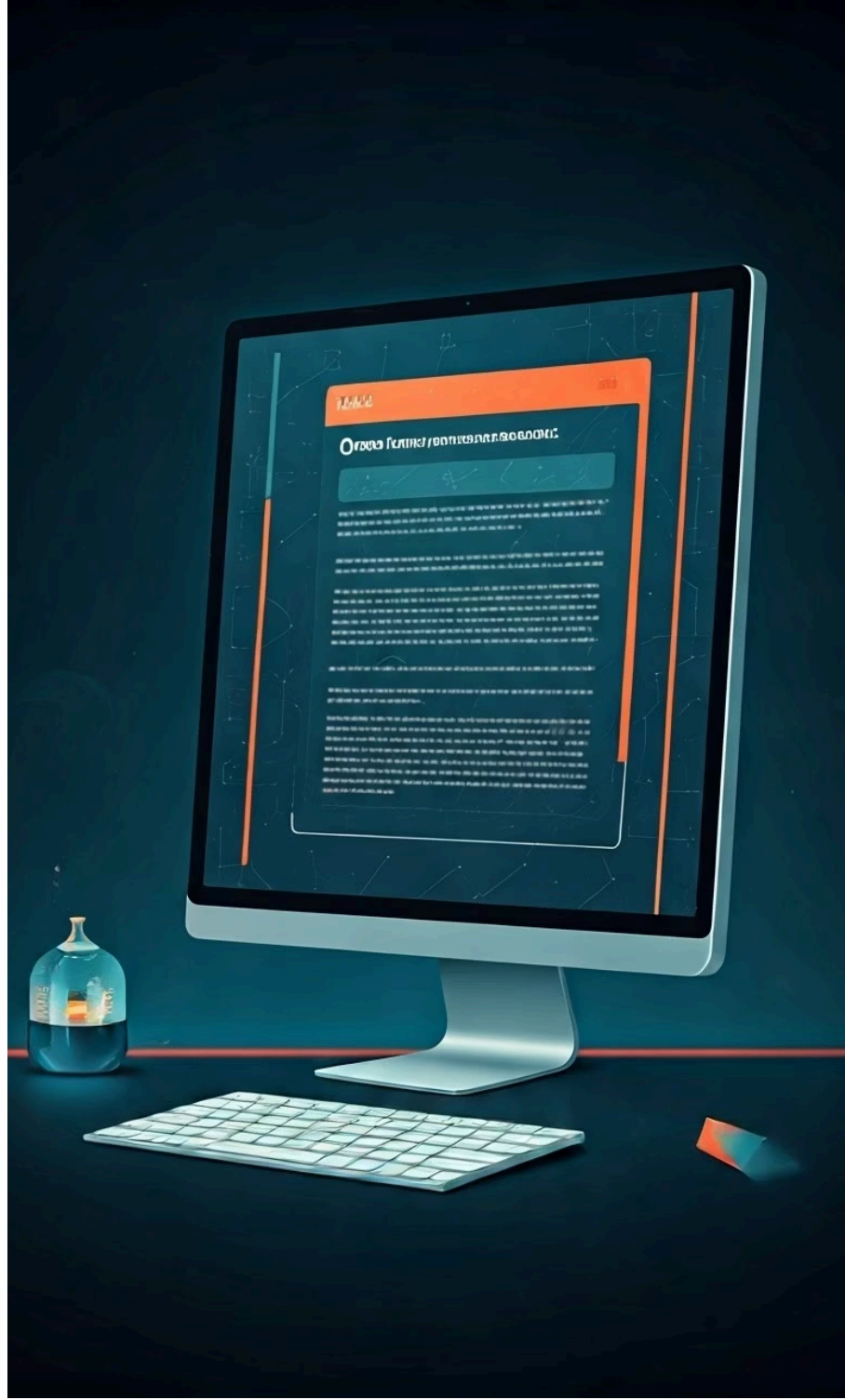
   The relevant context information is synthesised.

3. **Reformulation**

   A new query that integrates the compressed context is created.

4. **Optimised Retrieval**

   The compressed query is used to obtain more precise results.



Query Compression: Example

[... previous conversation about language training...]

Original Query

"What is the schedule of the course"

Derived Sub-queries

"What is the schedule of the C1 English course taught at the Puerto Real Campus during the first semester of the 2024/2025 academic year?"

Query Routing: Query Routing

1 **Relevance**

Ensures that the information retrieved is relevant for generating responses.

2 **Efficiency**

Directs the query to the most appropriate data sources, optimising resources.

3 **Filtering**

Can identify queries that the system should not respond to, improving security.

4 **Hybrid Strategies**

Combines keyword-based and semantic routing (using an LLM) for greater precision.

Query Routing: example

Original Query

"What cultural events are organised at the University of Cádiz?"

Router

The router decides to direct the query to a specific repository with data on events organised by the University of Cádiz, rather than searching the entire web

Query routing: system prompt

"Based on the user query, determine the most suitable data source(s)
to retrieve relevant information from the following options:
{{options}}
It is very important that your answer consists of either a single number
or multiple numbers separated by commas and nothing else!
User query: {{query}}"

Post-Retrieval Techniques

Post-Retrieval Techniques

Rerank

Reorganises the results to prioritise the most relevant ones.

Fusion

Combines results from multiple queries, resolving contradictions and creating coherent responses.

Summary

Synthesises and highlights the essential information, reducing token overload.

Rerank: Reordering the results

Similarity does not guarantee relevance: although semantic search already offers the results in order, it does not mean that the results it produces are the most relevant

"Lost in the middle": LLMs, like people, tend to focus on the beginning and end of texts.

Consider specific metadata: document popularity, source authority, temporal relevance, user preferences.

The aim is to introduce diversity in the results to avoid repetitions.

There are algorithms (TD-IDF, BM25) and specialised AI models for these tasks (Cohere, Jina, etc.)

Rerank: Example

Semantic retriever results

Reranking results

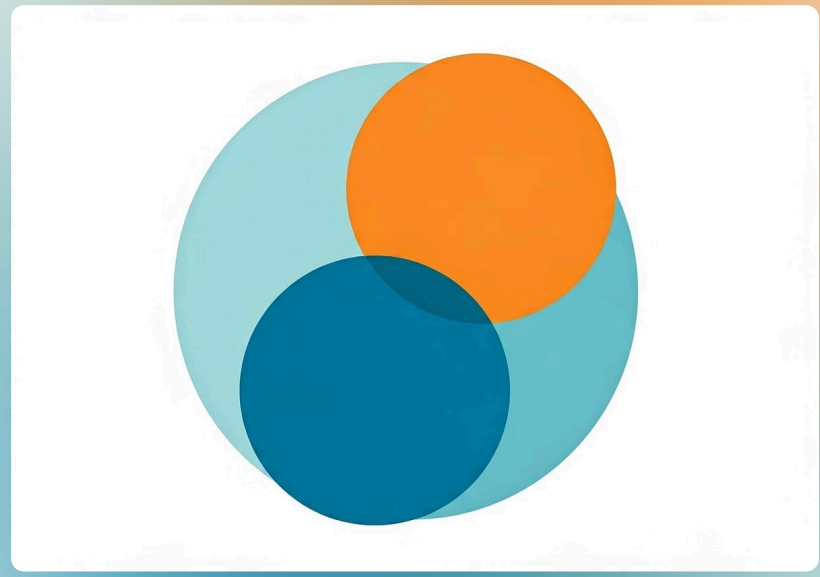
Fusion: Unifying Responses

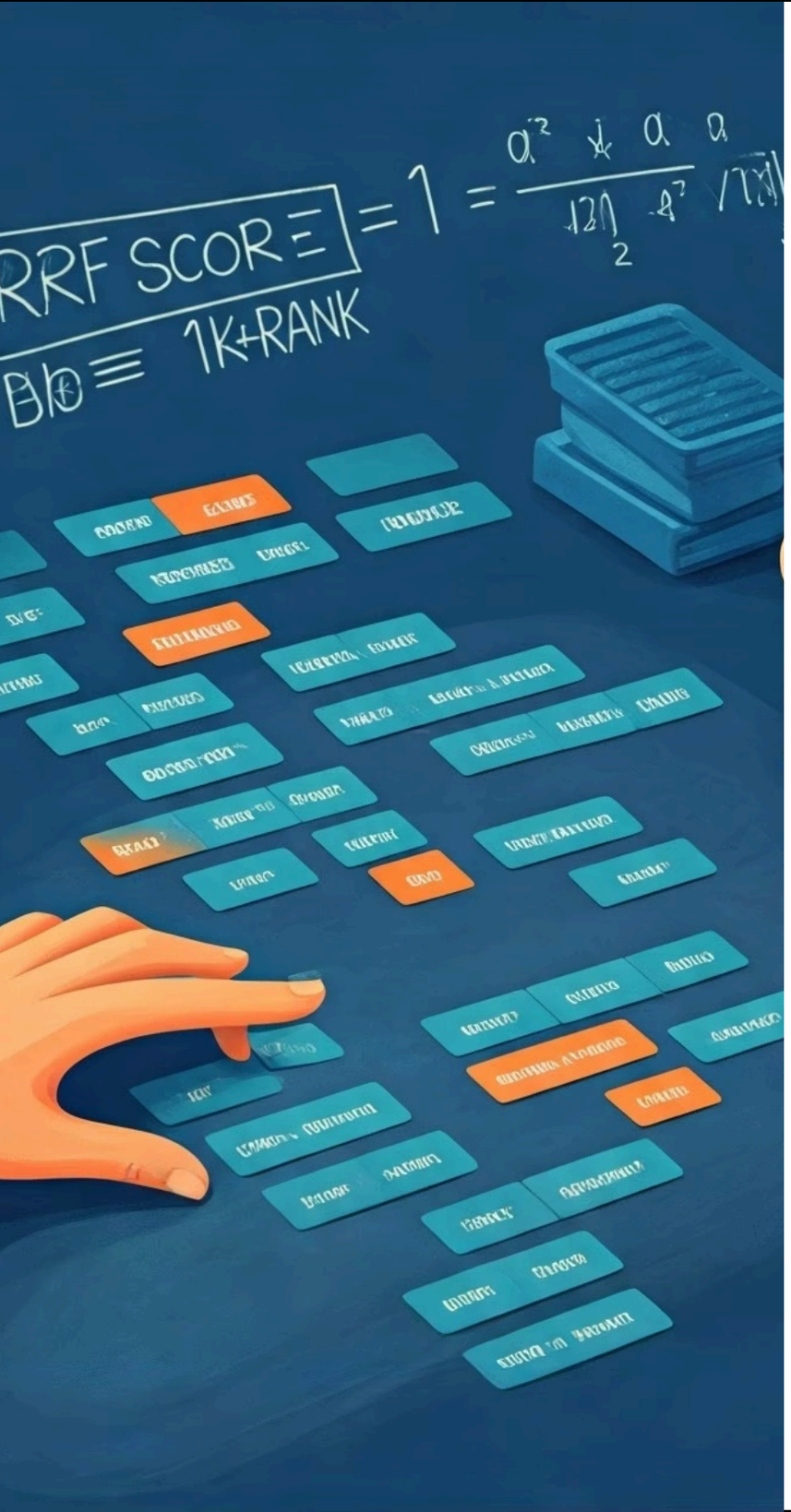
Combining Results

When using query expansion, we need to combine results to obtain complete and coherent answers.

Reciprocal Rank Fusion (RRF)

This technique assigns relevance scores to documents from different lists, then orders them to create a unified response.





Fusion: example

Semantic retriever results

	desarrolla software, aprende inteligencia artificial y redes."	
2	Documento G: "El Grado en Telecomunicaciones de la UCA abarca tecnología de redes, transmisión de datos y telecomunicaciones."	9.0
3	Documento H: "El Grado en Ingeniería Electrónica y Automatización ofrece formación en control de procesos industriales en la UCA."	8.5
4	Documento I: "La UCA imparte el Grado en Diseño Gráfico, con especial	6.1

	enfoca en redes y comunicaciones."	
2	Documento B: "El Grado en Ingeniería Electrónica Industrial en la UCA incluye automatización y control."	0.82
3	Documento C: "El Grado en Ingeniería Informática en la UCA cubre programación, redes y sistemas operativos."	0.80
4	Documento D: "El Grado en Diseño Gráfico en la UCA combina creatividad y	0.75

Fusion results

Documento	RRF Score	Ajuste final
Documento C / Documento F (Ingeniería Informática)	0.0323	**1er lugar**
Documento A / Documento G (Ingeniería Telecomunicaciones)	0.0325	2do lugar
Documento B / Documento H (Ingeniería Electrónica Industrial)	0.0320	3er lugar
Documento D / Documento I (Diseño Gráfico)	0.0312	4to lugar
Documento E / Documento J (Estudios Ingleses)	0.0306	5to lugar

Summary: Condensing contextual information

Distil, abstract and compact the retrieved texts, highlighting the key concepts and facts.

The aim is to avoid providing the LLM with superfluous or confusing information and reducing the number of tokens.

This helps the model to better understand the information and generate more accurate responses.

Summary: example

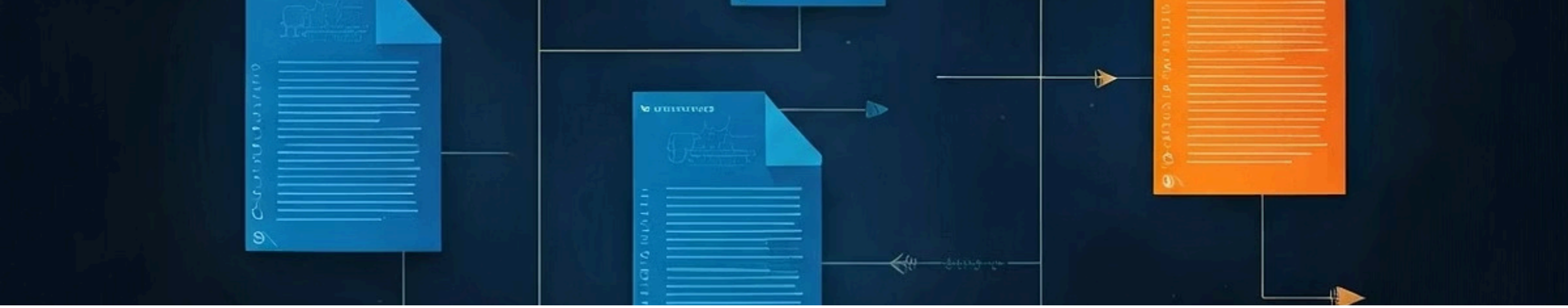
Results obtained

Entry to the LLM

Other advanced techniques

Other advanced techniques

- Summary embedding
- Parent document retrieval
- Multimodal RAG
- Hypothetical Questions Embedding
- Hypothetical Document Embedding
- Semantic chunking
- Contextual retrieval
- ...



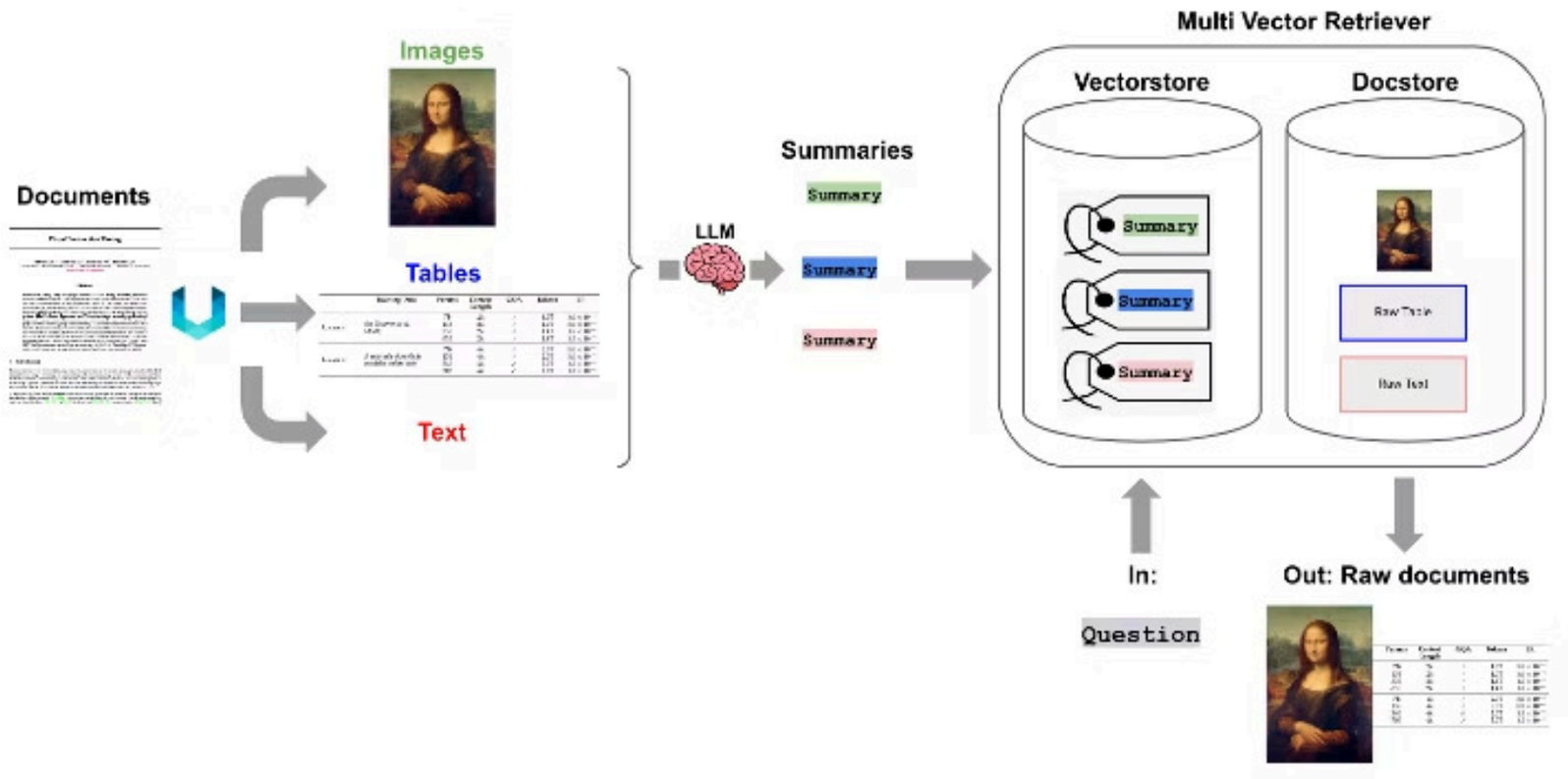
Summary Embedding + Parent

- When dividing documents for retrieval, there are often conflicting desires:
- The quality of **retrieval** improves when embeddings are generated from shorter texts. If they are too long, the embeddings may lose the full semantics of the text.
- The quality of **generation** improves when a long and detailed context is available, as this carries more semantic relationships and concepts. If we use short texts, the responses of the LLM may be less accurate due to lack of sufficient information.
- Documents, in addition to **text**, may include other types of content.

https://python.langchain.com/docs/how_to/parent_document_retriever/

Multi-modal RAG

Documents, in addition to text, can include images and tables, which can enrich the knowledge of the LLM.



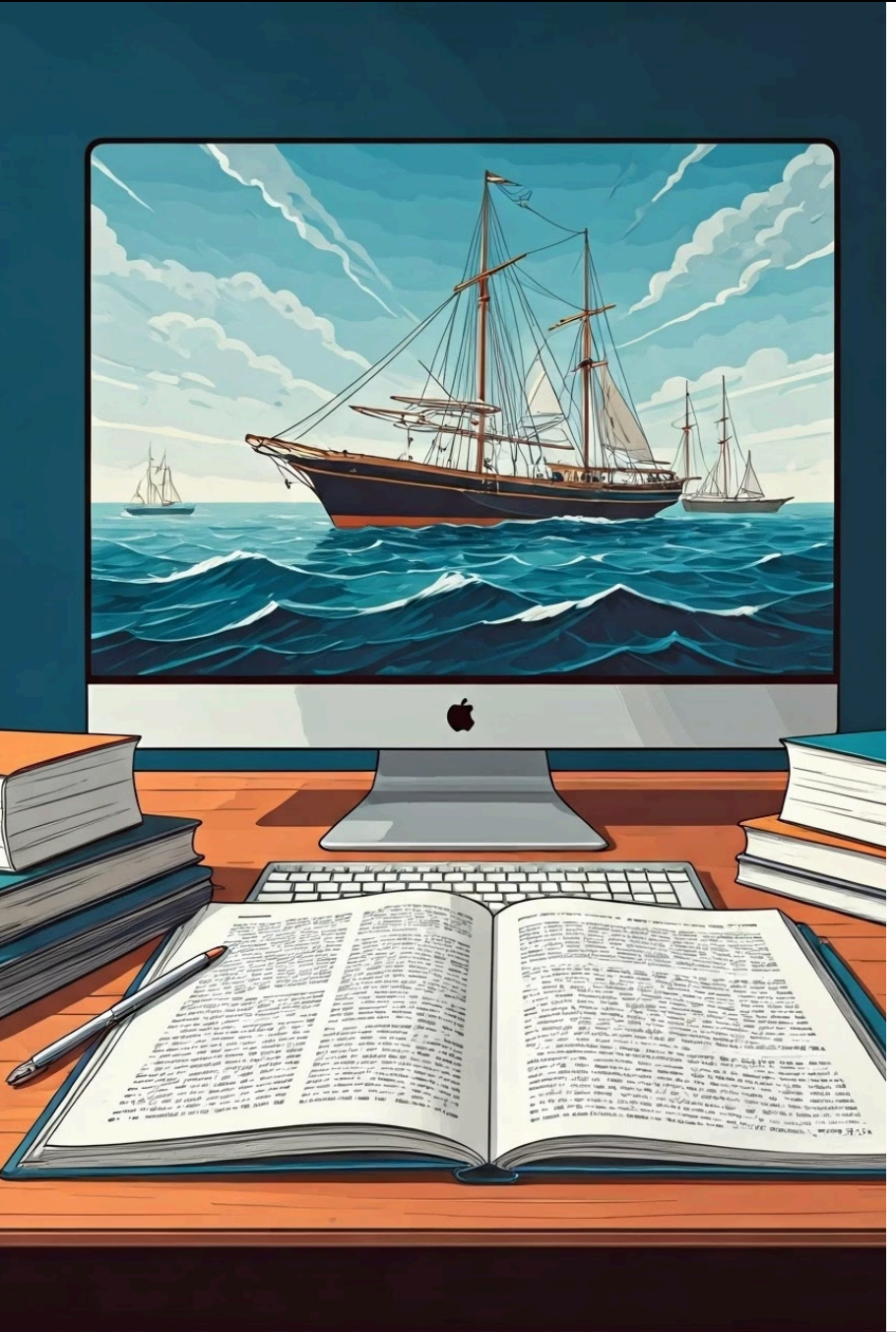
https://blog.langchain.dev/semi-structured-multi-modal-rag/

Hypothetical Document Embedding (HyDE)

Document similarity search can fail if the vector representation (embedding) of the query is not sufficiently similar to the relevant documents.

This technique generates a hypothetical document, which is then used to perform a similarity search in the knowledge base. This helps to find more relevant information and generate more accurate responses.

https://js.langchain.com/docs/integrations/retrievers/hyde/



Example of HyDE

Original Query

"How does the increase in ocean temperature affect the biodiversity of coral reefs?"

Generated Document

> "Global warming has caused a sustained increase in ocean temperatures, which negatively affects coral reef ecosystems. The rise in sea water temperature accelerates the phenomenon of coral bleaching, in which corals expel the symbiotic algae that nourish them, reducing biodiversity in these ecosystems. The loss of key species in the reefs has cascading effects that impact the entire marine food chain."

Summary

This presentation covered the evolution of information in the field of artificial intelligence (AI), specifically in the context of information retrieval and generation.

We explored how RAG, a technology that combines information retrieval with text generation, revolutionised the way machines understand and interact with knowledge.